UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Desarrollo integral de solución de ingeniería para un cliente real

Enrique Sánchez Bonet
Tutor: Fernando Sainz Paredes
Ponente: Álvaro Ortigosa Juárez

Junio 2017

Desarrollo integral de solución de ingeniería para un cliente real

AUTOR: Enrique Sánchez Bonet
TUTOR: Fernando Sainz Paredes
PONENTE: Álvaro Ortigosa Juárez

Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2017

Dedicado a mis padres,
Gracias.

"The men who go first are rarely popular with those who wait for the wind to blow"
Kaj Birksted

Resumen (castellano)

En este Trabajo Fin de Grado se analiza detalladamente el proceso íntegro del desarrollo de una aplicación web a medida para un cliente real. El proceso abarca desde la primera reunión con el cliente, donde estos exponen las necesidades que tiene para su negocio y donde se comienza a encaminar el proyecto, pasando por las fases de diseño y codificación, para finalizar con las pruebas finales por parte del cliente e inicio del mantenimiento.

El cliente tiene un negocio inmobiliario, centrado en la venta de casas de lujo. Su modelo de negocio era el tradicional. No habían dado el salto a las TIC. Acudieron a nuestra empresa con una idea sencilla, pero nada fácil de llevar a cabo, crear un portal inmobiliario de casas de lujo. Pero éste se conseguirá en la fase final del proyecto. La fase inicial, la cual se aborda en este TFG, consiste en crear una plataforma que sirva a la empresa del cliente para anunciar y vender sus propias casas.

La idea final de crear un portal de venta de casa de lujo resultó ser muy factible. Se han analizado las paginas ya existentes de venta de inmuebles de lujo y resulta que hay un nicho de mercado bastante potente.

La estrategia que se ha establecido para llegar a crear dicho por tal se divide en tres fases.

La fase inicial consiste en desarrollar la aplicación base para nuestro cliente, donde ellos puedan subir sus propias casas y que cumpla con los estándares de diseño, eficiencia y user experience actuales.

En la siguiente fase, que se prolongara más de un año, nos centraremos en el SEO y el SEM, así como en las redes sociales y la difusión. Una vez el sitio web comience a dar beneficios aumentando el número de inmuebles que vende el cliente, y el posicionamiento sea notable, se podrá iniciar la fase final.

Una vez demostrado el rendimiento y el posicionamiento de la aplicación, se comenzarán a anunciar inmuebles de terceros, garantizando la venta de estos a cambio de un porcentaje del beneficio.

La fase de desarrollo es crucial para crear unos cimientos fuertes sobre los que empezar a construir el resto del proyecto. El SEO y el SEM son otro factor crucial para alcanzar la meta final, y aunque no es objeto de análisis en este TFG, si se abordaran ciertas cosas básicas que afectan al desarrollo de la primera fase.

Abstract (English)

In this End-of-Degree Project it is analyzed with detail the entire process of development of a web application which is custom made for a real client. The process ranges from the very first meeting with the clients - where they expose the necessities they have for they business and when the process starts to be routed-, going through the different phases of design and codification, finalizing with the tests and closing the contract.

The client owns a real estate business, focused on the sale of luxury homes. Their business model was the traditional. They had not jumped into the ICTs until the moment, or at least not the way they desired. They came to our enterprise with an idea which was simple although not easy: creating a luxury real estate portal. This portal would be the last step in the path initiated with this project. The initial phase, which is treated in this project, consists on creating a platform for the clients' enterprise to both advertise and sell their properties.

The idea of creating a luxury real estate portal has proven to be, a priori, very feasible. The preexistent luxury real estate platforms were analyzed, and it turned out that there existed quite a powerful market niche.

The strategy established to reach the creation of this portal is divided into three phases:

The initial phase consists on developing the basis application for our clients -where they can upload their own houses- plus accomplishing their design, efficiency and user experience standards.

In the following phase, which extends to more than a year, we will focus on the SEO and the SEM, their social networks and the dissemination. Once the website starts giving benefits by increasing the number of properties our client sells, and once the positioning is noticeable, it is possible to initiate the final phase.

Once the application has reached a positioning enough to stand out from the rest of platforms, at least at a national level, the possibility of advertising on the website in exchange of a little percentage of the benefit will be given to third-party firms.

A good development is crucial to create some powerful foundations above which start building the rest of the project. The SEO and the SEM are another fundamental factor to reach the final goal, and although it is not an object of analysis in this project, some basic related notions are addressed given that they affect the development of the first phase.

Having said all this, this End-of-Degree project also has an utility beyond the success of a business project. The concepts and ideas here exposed could be useful for anyone who is willing to start a web project, without the availability of resources that a big enterprise can offer, being then able to compete with them. The chosen technology to develop this project reduces to the maximum the loads in terms of time, economic resources and working personal, allowing the creation of applications of equal or better quality than those developed by using traditional technology.

Palabras clave

Angular, Meteor, AngularJS, MeteorJS, MongoDB, Mongo, SASS, LESS, Full-Stack, StackMean, SEO, SEM, Material, AngularMaterial, MVC, Web3.0, Ingeniería del Software, Sistemas Informáticos, Servidor, NodeJS, JavaScript, Reactive, Responsive, Material Design.

Keywords

Angular, Meteor, AngularJS, MeteorJS, MongoDB, Mongo, SASS, LESS, Full-Stack, StackMean, SEO, SEM, Material, AngularMaterial, MVC, Web3.0, Software Engineering, Server, NodeJS, JavaScript, Reactive, Responsive, Material Design.

Agradecimientos

Quisiera dedicar este TFG a mis padres. Darles las gracias por el enorme esfuerzo que han realizado para ayudarme a llegar hasta aquí.

Me gustaría dar las gracias a mi tutor Fernando Sainz. Sin él, sin Afkar, este proyecto no habría sido posible.

Gracias a la Biblioteca de la EPS y a todo su personal. Ha sido mi segunda casa durante estos años.

A Pedro Ruiz y Julia Rubio, por demostrarme que son capaces de acompañarme sin dudar en cualquier aventura que se nos proponga. La suerte es la única esperanza de los que piensan que el éxito viene por accidente.

Dar las gracias a aquellos profesores que buscan innovar, que disfrutan enseñando y que se involucran al máximo con sus alumnos. Ellos son la esperanza para seguir quitando ladrillos del muro. Gracias por enseñarme que no todo eran matemáticas, bucles y punteros, que puede haber arte detrás de todo ello.

No concibo la palabra universidad sin Grapastagoras. Gracias a ellos he vivido los mejores años de mi vida. Ellos son la prueba fehaciente de que trabajando en equipo y ayudándonos, todo es mucho más fácil. Siempre nos quedará Villarosa.

Al Sargento Rida Jebaría, por enseñarme que lo importante son las personas.

A RITSI, en especial a Elena Ortega y Borja Andrino. Ejemplos a seguir como estudiantes, representantes académicos y sobre todo como amigos.

A todos aquellos que en algún momento han tomado parte en mi vida, sigan o no presentes, estén cerca o en otro continente. Sin ellos tampoco sería quien soy ahora.

Mención especial a María Iraizoz, la serendipia personificada, gracias por haber estado ahí todos estos años y sobre todo gracias por estar ahí ahora.

La verdad no es fácil escribir este tipo de cosas, son demasiados nombres como para mencionarlos a todos, demasiadas experiencias como para elegir una. Aun así, hay dos nombres que son los primeros que me vienen a la mente cuando echo la vista atrás y que por ello he dejado para el final. Mis hermanos Recio y Ovejero, la lucha de alfas. *Éramos tres almas perdidas nadando en la misma pecera, año tras año caminando sobre el mismo viejo suelo, donde encontramos los mismos viejos fantasmas, ojalá estuvieran aquí.*

Índice

Índice de figuras

1.Introducción

1.1.Marco del proyecto

La venta de inmuebles de lujo hoy día es un negocio muy rentable. Quizás la situación política económica y social de nuestra sociedad, que pasa por momentos difíciles, lleve a pensar que este tipo de productos no esté pasando por su mejor momento. Pero nada más lejos de la realidad. Lo inmuebles de lujo, así como otros productos de gran opulencia, no se ven afectados del mismo modo que el resto de productos. Es más, en épocas de crisis este mercado se ve beneficiado en sus operaciones.

Por otro lado, haciendo un análisis de la oferta de aplicaciones web de venta de inmuebles de lujo, se ve rápidamente que no es un sector muy explotado. Las webs existentes están bastante obsoletas, con unas RRSS nulas, una UEX no muy buena etc.

Y, por último, estos productos tienen un valor enorme, por lo que un mínimo porcentaje de los beneficios de una operación exitosa, ya supone una gran cantidad de dinero que apenas se verá reducido gracias al escaso coste de mantener una aplicación web. Con que se vendiese una casa más gracias a la aplicación en un año, ya se habrían amortizado de lejos las inversiones de ese año en la aplicación.

Por todo lo mencionado anteriormente, se llegó a la conclusión de que el desarrollo de una aplicación de venta de casas de lujo, combinando la fuerte experiencia del cliente en dicho sector, el uso de la última tecnología en cuanto a desarrollo web se refiere, y un buen SEO/SEM, es una gran oportunidad de irrumpir en el sector y marcar la diferencia con respecto a la oferta actual.

1.2.Motivación

Lanzarme al desarrollo de este proyecto no fue una decisión difícil. Hoy día existen numerosas herramientas que nos permiten crear una aplicación web rápida y fácilmente como Wordpress. Pero por otro lado han aflorado numerosos lenguajes que permiten dar a una web una serie de cualidades que hasta hace poco era muy complicado y costoso. Entre estas tecnologías se encuentran AngularJS y Meteor, usadas en el desarrollo de este proyecto

Dado que la mayoría de webs a día de hoy están desarrolladas con las herramientas mencionadas anteriormente, Wordpress y similares, una web desarrollada con una tecnología como Angular marca la diferencia rápidamente en cuanto a funcionalidad, diseño o posicionamiento. Además, es una tecnología nueva, en auge, desarrollada por Google.

Por todo ello decidí embarcarme en este proyecto.

Además, tendría la oportunidad, sin si quiera haber acabado la carrera, de llevar un proyecto por mi cuenta, lo cual me aportaría una formación y conocimientos cruciales para en un futuro poder emprender proyectos por mi cuenta. No todo es programar.

1.3. Objetivos

Por lo tanto, el objetivo de este proyecto es desarrollar la primera fase del proyecto. Crear una aplicación WEB totalmente funcional que pueda servir de base para el futuro portal inmobiliario de lujo. Cumpliendo con las exigencias del cliente, así como diseñar el proyecto de tal modo que permita explotar toda la ventaja que Angular y Meteor ofrecen a la hora de desarrollar una aplicación WEB.

2.Estado del arte

2.1.Tipos de inmuebles

Primero de todo hay que definir que es un inmueble. Un inmueble es todo aquello que no puede ser separado del lugar donde se halla.[1] Por todos es sabido la importancia económica y social que este mercado tiene en nuestra sociedad. Hay multitud de tipos de inmueble según su naturaleza, pero no es esta clasificación la que interesa en este caso.
Los inmuebles de lujo son el objetivo de este proyecto.

*2.1.1.*Inmuebles de lujo

Los productos de lujo no se rigen por las leyes normales del mercado, sino que parecen ir en contra de estas. Según la consultora Ernst and Young en su informe de 2016 *“The luxury and cosmetics financial Facebook”*[2] *la* industria del lujo ha aumentado en un 13% durante el 2015, en plena crisis. Según afirman también en este informe, al igual que ha sucedido en el resto de sectores, las nuevas tecnologías han cambiado radicalmente la forma de hacer negocio en este sector.
Hay un factor crucial en este tipo de comercio, que lo hace tremendamente rentable con respecto a otros productos, su precio. Debido a los altos precios, y al poco coste que conlleva desarrollar una aplicación web, con unos porcentajes de beneficio ínfimos con respecto al precio del producto fácilmente se consigue amortizar el gasto y obtener grandes beneficios.

2.2.Portales Inmobiliarios

En el marco inmobiliario español, este sector no está siendo muy explotado, o al menos no acorde a su potencial.
Según un estudio publicado por Idealista, un 81% de los europeos tiene internet como primera opción a la hora de comprar su casa y más de un 75% tienen éxito en esta búsqueda. España no se desmarca de esta tendencia, siendo internet la herramienta principal del 65% de los españoles a la hora de comprar un inmueble. [3]
Existen varios portales de compra-venta de inmuebles muy potentes a nivel nacional e internacional que analizaremos a continuación.

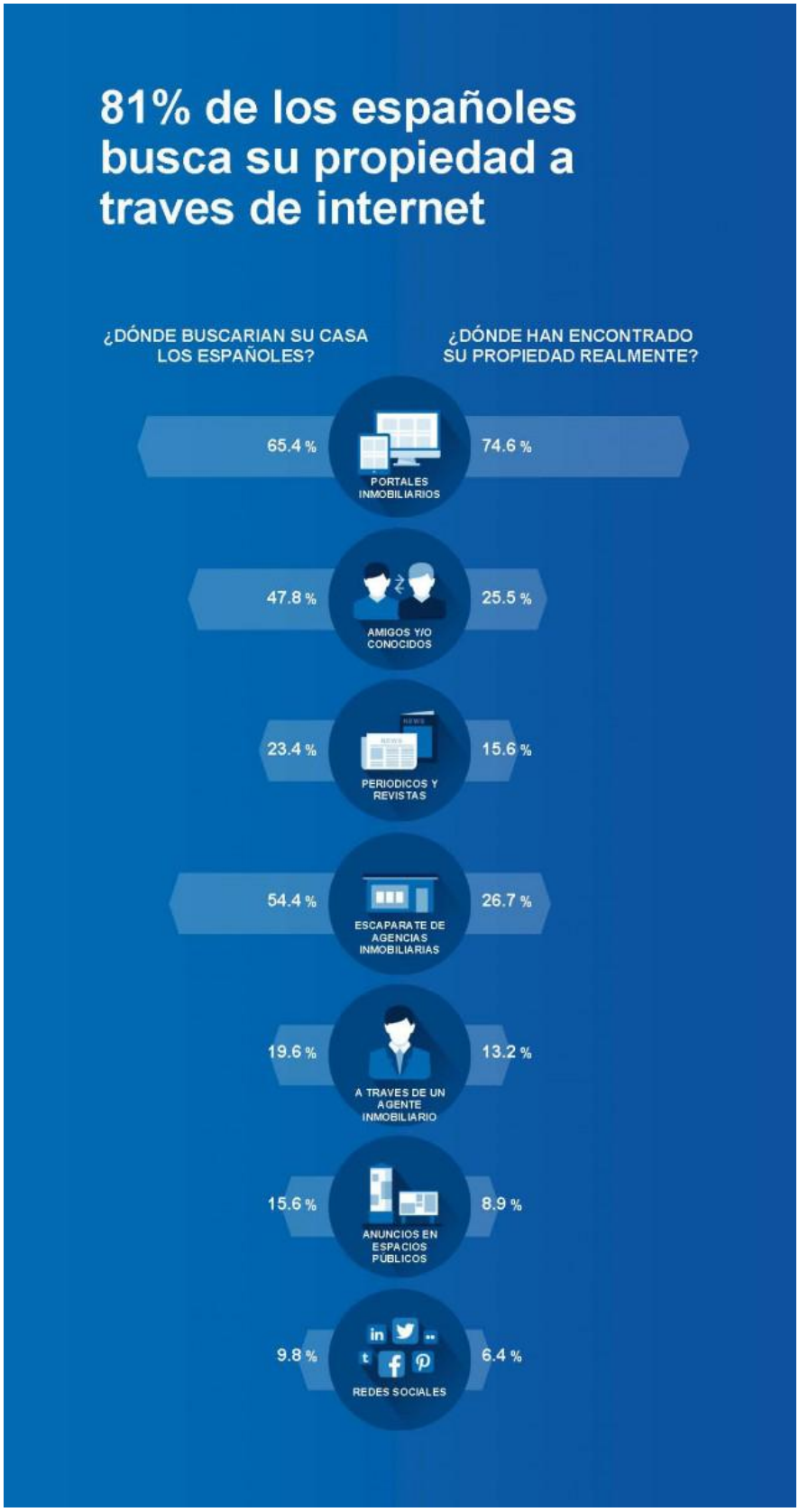


Ilustración 1 Estadísticas Idealista

[3]

2.2.1. Portales estándar

Son los portales principales. Anuncian inmuebles de todo tipo, tanto como para compra-vente como alquiler.
En estas plataformas también se anuncian muchos inmuebles de lujo, pese a que no es su nicho de mercado principal.
Entre los principales en España destacan cuatro, en función de su oferta de inmuebles y valoración de los usuarios.

- **idealista.com**

  Quizás el portal por referencia en España. Tiene la oferta más amplia y se pueden encontrar inmuebles de todo tipo. Permite filtrar por “comprar”, “alquilar” o “compartir. También permite filtrar la búsqueda mediante todo tipo de parámetros. Vivienda, vacacional, habitación, obra nueva, oficinas, terrenos etc. Sin duda es el referente de este mercado.
  En este portal operan tanto particulares como gran número de inmobiliarias. [4]

- **fotocasa.es**

  Este portal destaca también por su simplicidad. Uno de sus puntos fuertes es que permite buscar inmuebles cuyos precios estén actualizados. Su buscador por ubicación también es bastante “user friendly”. [5]

- **TuCasa.com**

  No es tan completa como Idealista pero destaca por su sencillez de uso. Es una plataforma muy agradable de manejar, de ahí que esa una alternativa muy popular por los usuarios. [6]

- **yaencontre.com**

  Posee un buscador por zona muy interesante, así como una opción para calcular la hipoteca asociada a un piso. [7]

*2.2.2.*Portales enfocados en el lujo

- **luxuryestate.com**

  Sitio de referencia en cuanto a compra venta de casas de lujo a nivel mundial. Posee la mayor oferta de inmuebles de lujo de toda la web.[8]

- **luxhome.es**

  No destaca por su diseño, pero está especializada en las zonas de Madrid e Ibiza. Es una de las empresas con mayor experiencia en el sector.[9]

- **ambassador.es**

  Diseño muy elegante pese a que la tecnología de esta web está algo anticuada. Es también una empresa con una gran trayectoria en el sector.[10]

2.3.Tecnología para el desarrollo web

Hoy día sería imposible definir el modo de vida de la sociedad si eliminásemos la palabra Web del diccionario. Web, Conjunto de información que se encuentra en una direcciónn determinada de internet. Ciertamente es muy difícil, casi imposible, encontrar alguna tarea cotidiana en cuyo proceso, en algún punto, no esté implicada la Web.
Mucho ha cambiado desde que Tim Berners-Lee y sus colegas del CERN creasen la primera página Web en 1991. En aquel entonces, solo un pequeño puñado de universitarios tenían acceso a ella [11].
Actualmente, hemos pasado de esa simple pero genial página plana con fondo blanco e hyperlinks en azul de 1991 a bases de datos accesibles desde cualquier lugar (cloud), sitios Web capaces de dar servicio a millones de personas al mismo tiempo, Inteligencias artificiales siendo entrenadas dio a día por miles de usuarios reales, Wikipedia etc.

El termino Web 2.0, o Web social, es un término ya arcaico. La evolución de la interacción de las personas en internet se encuentra hoy en un escenario muy diferente al existente en 2005 con el boom de las redes sociales [12].

La Web 3.0, accesible para cualquier persona sin depender de que dispositivo utilice. Una Web que va más allá del simple hecho de compartir información, sin que esto implique un beneficio para el usuario, manteniendo un intercambio de información seguro y responsable.

Está diseñada de una forma más humana, más centrada en las personas. Facilidad de uso, intuición, tiempos de respuesta, eficiencia, responsabilidad con el medio ambiente, en definitiva, una Web enfocada al bien común, al progreso y al desarrollo.

Bajo este contexto, han emergido numerosas tecnologías pretendiendo suplir las carencias que tenían los sistemas tradicionales de desarrollo web, HTML, CSS, php etc.

2.3.1. Front-End

En el desarrollo Front-End se aborda todo lo relacionado con la vista de la app, la parte visual. Como se muestran los datos y como interactúa el usuario con estos. Son numerosos los frameworks actuales orientados al desarrollo frontend, orientados todos a cumplir dos premisas, responsive y reactive.
Una aplicación responsive es aquella que se adaptar a cualquier dispositivo, manteniendo una UEX optima en cada dispositivo.
Un sistema reactivo es aquel en el que se actualiza la información sin necesidad de recargar la página completa. Si algún dato cambia en la base de datos, la representación de este en la vista cambiara inmediatamente. Conseguir esto con lenguajes tradicionales es una ardua tarea, pero con la tecnología actual bastante sencillo.

2.3.1.1. Angular

Angular[13] es uno de los principales frameworks que se utilizan actualmente. Es la que se ha usado en el desarrollo en este proyecto en la parte Front-End.
Angular nos permite desarrollar aplicaciones de calidad, sencillas con unos recursos muy limitados. Su punto fuerte es el Data Binding. Angular implementa de serie esta propiedad, la cual consiste en que la vista se actualiza automáticamente en cuanto el controlador se ve modificado. Sin retardos ni molestas pantallas blancas para el Usuario.
Angular se basa en el patrón Modelo Vista Controlador, ideal para aprovechar al máximo el Data Binding.
Otra gran cualidad de Angular es que está basado en JavaScript puro, no se necesita importar dependencias de terceros, lo cual hace a Angular muy sencillo de comprender y programar.
Angular también permite extender los tags de HTML clásicos creando componentes (directivas), lo cual permite reutilizar

el código enormemente haciendo muy Modular. La programación en angular se basa en la creación de componentes independientes y modulares que poseen cada uno su propia vista y controlador.
Por último, permite inyectar dependencias, lo cual permite declarar como la aplicación está “cableada” sin la necesidad de un main.js el cual suele ser origen de muchos problemas ya que se convierte con mucha facilidad en el “cajón desastre” del código.
Otra ventaja saliendo del ámbito de la programación es que Angular está fuertemente impulsado por Google, por lo que cualquier aplicación desarrollada con Angular parte con una ventaja competitiva a nivel posicionamiento con respecto a otras aplicaciones.

2.3.1.2. Bootstrap

Framework liberado por Twitter. Sirve para maquetas de forma rápida y sencilla la vista de una aplicación Web[14]. La clave es el grid basado en 12 columnas. Cada elemento consta de 12 unidades, así este puede ser divisible entre 2, 3, 4 y 6. Esto Permite tener la vista de cualquier aplicación muy bien organizada y limpia además de adaptarse a cualquier tipo de dispositivo.
Es un framework con una curva de aprendizaje muy lapida y es opensource.

2.3.1.3. React

React[15] es otra de las alternativas para el desarrollo de la vista de nuestro proyecto. Liberado por Facebook, se ha utilizado para desarrollar aplicaciones tan importantes como Instagram. Es un muy buen complemento para Angular, pudiendo dejar este para el manejo de las vistas y Angular en el controlador.

2.3.1.4. Ember

Ember[16] es similar a los frameworks anteriores. Quizás su característica más destacable es que ofrece un ingenioso sistema de control de versiones. Se te va avisando de las pequeñas actualizaciones de porciones de código que deberán migrarse, hasta que finalmente, cuando sale la

release, este deja de funcionar. Así te obliga a ir actualizando el código poco a poco.

2.3.1.5. SASS y LESS

SASS[17] y LESS[18] son pre compiladores de CSS. Nos permiten reescribir el código CSS de una forma más sencilla y fácil de leer. Además, añaden características muy deseables como herencia u operaciones y funciones, las cuales no están disponibles en CSS3 nativo.

2.3.2. Back-End

La parte Back-End es la otra cara de la moneda en el desarrollo web. En esta rama siguen persistiendo con más facilidad las tecnologías tradicionales. Es habitual ver coexistir Front-End desarrollado en JS comunicándose mediante APIs con modelos desarrollados en PHP, Java etc. Aunque cada vez cobran más fuerzas nuevas alternativas como Node o Meteor, que apuestan por compartir lenguaje en el lado cliente y en el servidor.

2.3.2.1. NodeJS

Node.js[19] es el mayor ecosistema de librerías de software libre que existe. Está basado en JavaScript y es Opensource. La clave de Node es utilizar JavaScript tanto en el lado del cliente como en el lado del servidor.
La principal ventaja de Node frente a sistemas de servidor tradicionales como Java o PHP es la eficiencia. Estos servidores por lo general generan un hilo para cada conexión de unos 2MB. Por lo que, si se necesitase atender a 16.000 usuarios, cifra no muy alta para aplicaciones web grandes, necesitaríamos un servidor de 32GB de rama y ya iría apurado en recursos.
Node reinventa este sistema y elimina la creación de un hilo de ejecución para cada llamada. Cada conexión entrante dispara la ejecución de un evento en el servidor que será atendido por este. Un servidor de Node no se queda “colgado” ya que no se permiten los bloqueas ni las espera a los sistemas de E/S. Gracias a esto se pueden atender a miles de peticiones concurrentes con unos recursos de memoria muy limitados.

El ecosistema contiene una infinidad de paquetes desarrollados por la comunidad para suplir casi cualquier necesidad de un desarrollador, lo que agiliza enormemente el desarrollo de un proyecto, permitiendo al desarrollador reutilizar numerosos módulos externos y completamente libres.
Por último, cabe destacar que el motor de Node es tremendamente rápido, el motor V8 desarrollado por Google

2.3.2.2. Meteor

Meteor[20] es un framework de JavaScript que funciona sobre Node. La mejor forma de explicar las cualidades de meter es mediante un ejemplo. Al abrir la misma carpeta en dos ventanas diferentes del explorador de archivos del sistema, ¿Que ocurre si borramos algún archivo en una de las ventanas? En efecto, dicho archivo desaparece también de la otra ventana. Esta cualidad es la que meter ha querido implementar en los sistemas cliente servidor. Meteor abandona la antigua idea de que la comunicación cliente servidor es algo ocasional, convirtiéndola en algo constante. Para conseguir esto, un servidor de meteor mantiene una lista de los clientes que están conectados al servidor, haciendo uso de algunos de los recursos de este. Si alguno de estos recursos cambia, meter notifica el cambio a todos los clientes conectados, actualizando la información sin necesidad de mandar una nueva petición por parte del cliente.
Otra funcionalidad esencial que aporta meteor es un novedoso sistema de base de datos. Ahora existen dos bases de datos, una local y una en el servidor. Cada vez que el cliente desea modificar información de la base de datos, estos cambios. se hacen sobre la base de datos local, la cual se comunica a continuación con la base de datos remota en un segundo plano, así se puede continuar trabajando con la impresión para el usuario de que la comunicación ha sido instantánea. En caso de ocurrir algún error, la base de datos local y remota resuelven el problema, casi siempre, sin efecto alguno en la experiencia del usuario.

2.3.2.3. MongoDB

MongoDB[21] es una base de datos orientada a documentos, es decir, es No Relacional. No existen las tablas ni las consultas SQL.

Mongo guarda los datos en documentos, y la información no tiene por qué seguir una estructura. Se puede tener una colección de objetos y que estos tengan propiedades diferentes entre sí o incluso propiedades con diferentes tipos de datos.

Los datos se guardan en objetos JSON, lo que nos ofrece una compatibilidad y simplicidad enorme si utilizamos alguna de las tecnologías anteriormente mencionadas, todas en JS.

MongoDB es muy útil si se quiere desarrollar un proyecto altamente escalable.

Las limitaciones de Mongo con respecto a las bases de datos relacionases tradicionales es la imposibilidad de realizar transacciones y consultas con JOIN, aun así, lo compensa con una sencillez y potencia enormes.

2.3.3. FullStack

En los últimos tiempos, las grandes empresas cada vez demandan más desarrolladores FullStack. Un desarrollador FullStack es aquel que conoce bien todo lo relacionado con el Front-End y el Back-End. Es un perfil muy generaliza con mucho conocimiento técnico. Es decir, un desarrollador FullStack debe ser capaz de desarrollar aplicaciones desde cero el solo.

El kit StackMean es que se ha explicado hasta ahora, basado en MongoDB, Angular y Node.

Otras cualidades imprescindibles para un desarrollador FullStack son:

- Conocer soluciones Cloud.
- Control de versiones, en concreto GIT.
- Métricas y Google Analitycs.
- Seguridad.
- Posicionamiento SEO.

2.4.Tendencias de Diseño

El diseño es quizás el factor determinante del desarrollo Web. Un buen diseño marca la diferencia. Está constatado que un usuario tolera mejor una página con un rendimiento no muy bueno y con un diseño genial que al revés.
Los usuarios valoran lo que les entra por los ojos, por lo que una página “fea” es un fracaso. El fin último de una Web es que sea usable por el usuario.

*2.4.1.*Flat

El minimalismo es una tendencia muy popular desde hace ya bastantes años, y el diseño Web no es una excepción. El minimalismo en el diseño Web dio lugar al “flat design[22]” o diseño plano. Esto además contribuir a mejorar el rendimiento de la web ya que las imágenes y los recursos son simples y ocupan poco espacio.

*2.4.2.*Material

El flat design ha ido avanzando, impulsado por Google en gran medida, y a día de hoy es muy popular una de sus variantes, el “material design”. Este está basado en el diseño plano, pero añadiendo sobras, degradados y animaciones entre sus componentes. Son muchos los frameworks orientados a este tipo de diseños. Por ejemplo, Angular Material, similar a Bootstrap, pero enfocado en Angular.

2.5.Conclusiones

Tras todo lo expuesto anteriormente, se puede deducir que colaborar con una inmobiliaria especializada en casas de lujo es una gran oportunidad. En la que los beneficios por ambas partes pueden ser notables. Las TIC hoy día son imprescindibles para este tipo de negocio y es un mercado que generara muchos beneficios para todas las partes implicadas.
En cuanto a las tecnologías a usar, hay varios factores a tener en cuenta. Los recursos económicos y personales son bastante limitados, por lo que hay que encontrar una fórmula que permita desarrollar código de calidad de forma rápida y sencilla, permitiendo reutilizar código, con un diseño modular y escalable.
Además, el posicionamiento SEO es esencial en este modelo de negocio.

También ha de ser fácil desarrollar una Web con un diseño moderno y atractivo para el usuario.
El marco general del proyecto será NodeJS. Por su gran rendimiento en el lado del servidor y la amplia oferta de módulos disponibles para abordar cualquier aspecto del desarrollo de la aplicación.
El Back-End lo desarrollaremos en Meteor, ya que aporta de base unas ventajas competitivas muy fuertes con respecto a otras tecnologías.
En cuanto a la base de datos, por su sencillez y la posibilidad de crear modelos con objetos heterogéneos, será MongoDB.
En cuanto al Front-End, elegimos Angular1. Es imprescindible que la aplicación web sea 100% responsive y reactiva. Además, Angular también se ve beneficiada por Google en cuanto al SEO. Usaremos sobre angular el Framework Angular material, lo que aumentara enormemente la velocidad y la calidad de la maquetación.
Usaremos además SASS como lenguaje CSS.

3.Análisis

3.1.Introducción

En este apartado se expone en profundidad la fase de análisis del proyecto. Dado que va a ser una aplicación para un cliente real, esta fase es la más importante del proyecto, ya que una mala captura de requisitos tiene unas consecuencias desastrosas para las fases posteriores.
Se realizaron en total tres reuniones con el cliente, enfocadas en realizar la captura de requisitos. Tras esto se consiguieron definir unos requisitos sólidos, aprobados por ambas partes. Pese a esto, como es habitual en cualquier proyecto real, surgieron requisitos futuros, pero gracias al buen análisis inicial, y las cualidades de la tecnología utilizada, la cual aporta gran independencia de módulos y escalabilidad, los nuevos requisitos se añadieron sin problema.

3.2. Roles de usuario

En una primera fase solo habrá un tipo de usuario. El usuario final. Se ha contemplado la posibilidad de añadir un usuario administrador, para poder subir nuevas casas a la base de datos. Dicha mejora se realizará en una fase posterior del proyecto.

*3.3.*Casos de uso

Se busca una aplicación sencilla, minimalista, que no sobrecargue al usuario con muchas opciones y posibilidades.
Básicamente el usuario puede buscar casas aplicando una serie de filtros, consultar la ficha técnica de una casa en concreto y solicitar información sobre dicha casa.
Además, existirán una serie de opciones para las que el usuario podrá rellenar un formulario online que será enviado directamente al correo de la empresa.

RF-1 También existirá la opción de abrir la aplicación de correo estándar para redactar un correo personalizado

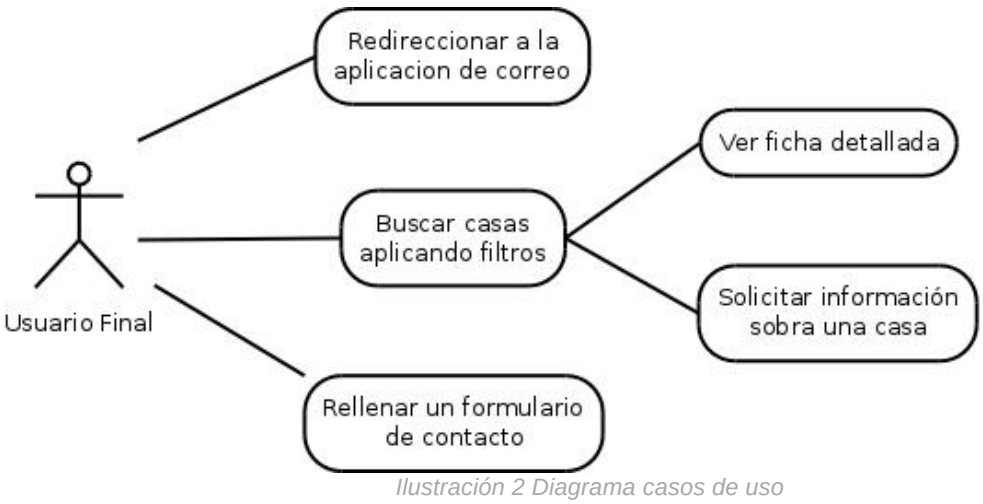


Ilustración 2 Diagrama casos de uso

*3.4.*Definición de requisitos

Los requisitos de la aplicación se dividen en dos grupos

- Requisitos funcionales
- Requisitos no funcionales

*3.4.1.*Requisitos funcionales

La numeración de los requisitos no sigue un formato estándar. Los requisitos asociados a una futura "task" a desarrollar por el programador están enumerados incrementalmente. Así por ejemplo el requisito 5.14, es el requisito (task) número 14 en el total de requisitos y está en el grupo 5 de requisitos.

RF-1 Se podrán filtrar casas con un buscador según una serie de parámetros.

RF-1.1. Filtrar por tipo de operación
- **RF-1.1.1** Compra
- **RF-1.1.2** Venta

RF-1.2. Filtrar por país
- **RF-1.2.1** España

RF-1.3. Filtrar por zona
- **RF-1.3.1** Madrid
- **RF-1.3.2** Ibiza

RF-1.4. Filtrar por área
- **RF-1.4.1** Barrio de Salamanca
- **RF-1.4.2** Chamberí
- **RF-1.4.3** Somosaguas
- **RF-1.4.4** Calle Ortega y Gasset
- **RF-1.4.5** Arturo Soria
- **RF-1.4.6** Nacional I. Ciudalcampo
- **RF-1.4.7** Nacional VI. La Florida

RF-1.5. Filtrar por tipo de inmueble
- **RF-1.5.1** Piso
- **RF-1.5.2** Dúplex
- **RF-1.5.3** Ático
- **RF-1.5.4** Chalet adosado
- **RF-1.5.5** Chalet independiente
- **RF-1.5.6** Estudio o loft
- **RF-1.5.7** Edificio
- **RF-1.5.8** Nave
- **RF-1.5.9** Local
- **RF-1.5.10** Oficina
- **RF-1.5.11** Terreno rústico o urbano
- **RF-1.5.12** Finca rústica o de recreo

RF-2 Se podrá rellenar un formulario Online de contacto

RF-2.6. Existirán varias secciones con una breve descripción en las cuales el usuario podrá rellenar un formulario

RF-2.6.1 Vendemos y alquilamos tu propiedad
RF-2.6.2 Alquilamos tu casa para eventos
RF-2.6.3 Reformas y decoración
RF-2.6.4 Trabaja con nosotros
RF-2.6.5 Inversores
RF-2.6.6 Contacto

RF-3 Se podrá acceder a la ficha completa de una casa, que contendrá la siguiente información.

RF-3.7. Galería multimedia compuesta de dos pestañas

RF-3.7.1 Video. Un video reproducible desde YouTube
RF-3.7.2 Carrusel de fotos

RF-3.8. Características generales de la propiedad

RF-3.9. Descripción detallada en formato texto.

RF-3.10. Solicitud de información.

RF-4 Un banner superior que deberá estar presente en todas las pantallas.

RF-4.11. En la parte izquierda aparecerá el logo de Cassona

RF-4.12. En la parte derecha habrá un botón que desplegará un menú que redirigirá a cualquier sección de la aplicación.

RF-5 Deberá haber una bottom bar, presente en todas las pantallas salvo en la home, en la que aparecerá lo siguiente.

RF-5.13. Aviso Legal

RF-5.14. Link a la home

RF-5.15. Página de contacto

RF-5.16. Redes sociales

RF-5.17. Copyright

RF-5.18. Correo

RF-5.19. Teléfono

RF-6 En la home aparecerá lo siguiente

RF-6.20. Un carrusel con las mejores fotos

RF-6.21. El buscador

RF-7 Con el resultado de las búsquedas se mostrará un mosaico con todas las propiedades que coinciden con esa búsqueda.

RF-7.22. Al pulsar sobre uno de los elementos se abrirá la ficha extendida.

RF-7.23. En la parte superior habrá un buscador con el que poder hacer una nueva búsqueda

3.4.2. Requisitos no funcionales

RNF-1 Debe ser totalmente responsive. La web debe visualizarse correctamente en cualquier tipo de pantalla.

RNF-2 Debe poder visualizarse correctamente en Safari, Chrome y Firefox en sus versiones actualizadas.

RNF-3 Se debe poder llegar a la ficha técnica de una casa en menos de 3 clics.

RNF-4 Debe tener un funcionamiento fluido en un dispositivo móvil de gama media, por lo que no debe consumir muchos recursos.

4.Diseño

4.1.Introducción

En este apartado se abordarán los aspectos más importantes en cuando al diseño de la aplicación.
Se expondrán 4 diseños diferentes los cuales permiten tener una visión global de la aplicación. Se analizará la estructura de la parte front, del servidor, de la base de datos y del funcionamiento de la aplicación.

4.2.Cliente

Se ha seguido el patrón Modelo Vista Controlador (MVC) [23]. Como se ha explicado anteriormente, este modelo es ideal para combinarlo con Angular, ya que este lenguaje aprovecha al máximo las características de este modelo.
Angular permite crear nuevos tags HTML por lo que se han creado componentes, por ejemplo, el mosaico de fotos o un elemento del mosaico de la foto.
Un componente puede estar formado a su vez por otros componentes. Una buena analogía es pensar que cada componente es un objeto, el cual puede estar formado por o formar parte de otros objetos.

Cada componente se compone de tres ficheros.
Un fichero HTML que proporciona la vista del componente.
Un fichero JavaScript que contendrá el controlador del componente, así como diversas funciones auxiliares. Entre la vista y el controlador existirá un binding de datos, de tal forma que cuando cambie algo en el controlador cambiará automáticamente en la vista.
Y por último un fichero SASS que contendrá el estilo. El fichero SASS es compilado en CSS. A continuación, se muestra un diagrama completo de los componentes de la aplicación.

4.2.1.Diagrama general de la aplicación

Afkar es el elemento general que contiene el nav superior, presente en todas las pantallas, y el fragmente con la pantalla seleccionada, ui-view. Hay 4 tipos de vista principal, home, propiedades verPropiedad e inversores/contacto/reformas etc.
Cada una de esas vistas a su vez contienen otras en su interior.

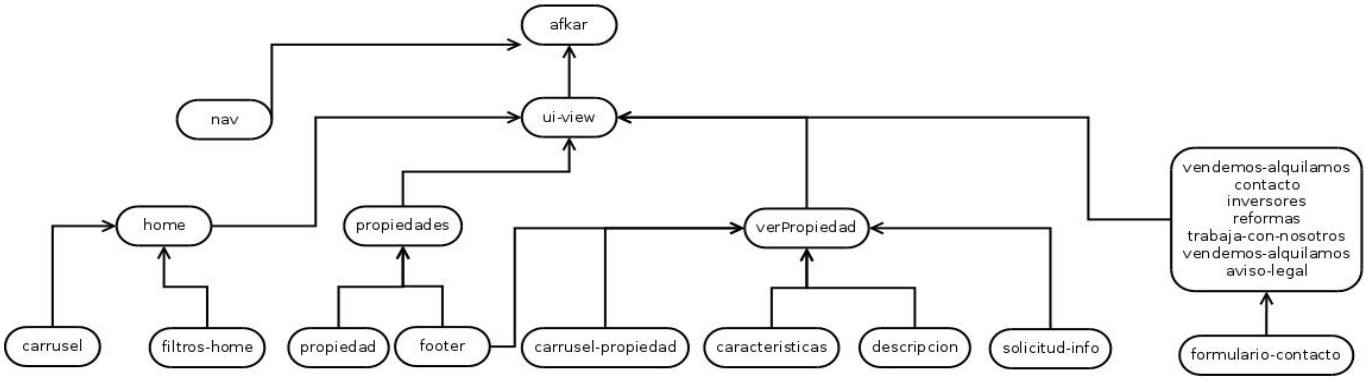


Ilustración 3 Diagrama general de la aplicación

4.2.2.Servidor

La arquitectura es la clásica de cualquier aplicación cliente servidor. En la siguiente imagen se ve claramente el flujo de las peticiones.

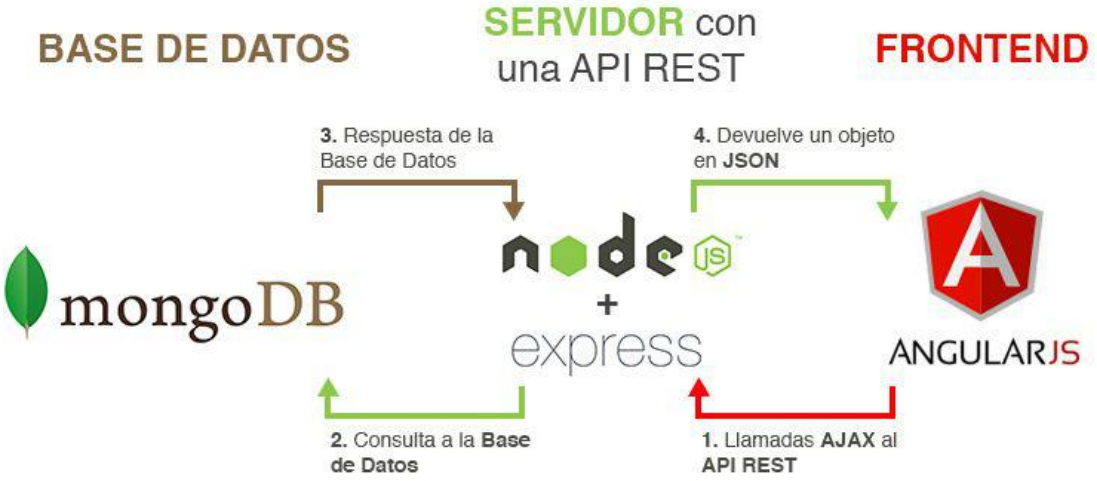


Ilustración 4 Arquitectura Cliente Servidor [24]

*4.2.3.*Filtros

```
export const Filtros = [
{
    id: 1,
    nombre: "modo",
    valores: [
    { id: 1, valor: "Venta" },
    { id: 2, valor: "Alquiler"  }
    ]
},
{
    id: 2,
    nombre: "localizacion",
    valores: [
    {
        id: 1,
        valor: "España",
        zonas: [
        {
            id: 1,
            valor: "Madrid",
            areas: [
            { id: 1, valor: "Barrio de Salamanca" },
            ...
            { id: 7, valor: "Nacional VI. La Florida"}
            ]
        },
        {
            id: 2,
            valor: "Ibiza",
            areas: [
            { id: 1, valor: "Marina Botafoc" }
            ]
        }
        ]
    }
    ]
},
{
    id: 3,
    nombre: "tipo",
    valores: [
```

Ilustración 5 Estructura de filtros

Los filtros son un simple array de objetos, con una serie de propiedades identificadas cada una con un id.

En el controlador del componente propiedades es donde están definidas las querys donde en función de las propiedades del filtro seleccionadas y los atributos de cada elemento de la base de datos, se devuelve un array que cumpla dichas cualidades. Destacar que como en MongoDB no existen los JOINS por lo que esta funcionalidad hay que implementarla a mano para filtrar correctamente.

4.2.4.Base de datos

Al ser una base de datos no relacional, no hay una estructura definida como tal. Se podría insertar en la colección un elemento con los atributos y tipos que se desee, que la aplicación seguirá funcionando correctamente. Aun así, se ha definido un formato estándar que deberán seguir todos los nuevos elementos que se introduzcan en la base de datos.

```
var nuevaPropiedad = {
                    _id: String,
                    idModo: int,
                    fotos: int, //numero de fotos
                    video: String,
                    idPais: int,
                    idZona: int,
                    idArea: int,
                    idTipo: int,
                    tam: int,
                    precio: int,
                    caracteristicas: String,
                    descripcion: String
          };
```

Lo filtros trabajan utilizando esta información para clasificar las casas. Si no se sigue esta estructura, los filtros no funcionaran correctamente y pueden quedar casas sin mostrar. La ventaja de esto es que es muy sencillo añadir un campo más y nuevos filtros. No hay que modificar nada más que esa estructura y todo seguirá funcionando correctamente.

4.2.5.Dependencias

Estas es el árbol de dependencias del proyecto. Esto son los paquetes de meteor que hemos instalado. Meteor contiene su propio repositorio de paquetes, al igual que node. Meteor a su vez es un paquete de node.

```
* New versions of these packages are available! Run 'meteor update' to try to
  update those packages to their latest versions. If your packages cannot be
  updated further, try typing `meteor add <package>@<newVersion>` to see more
  information.
MacBook-Pro-de-Quique:cassona-webapp quiquesanchebonet$ meteor list
es5-shim                            4.6.15  Shims and polyfills to improve ECMAScript 5 support
insecure                            1.0.7  (For prototyping only) Allow all database writes from the client
jquery                              1.11.10  Manipulate the DOM using CSS selectors
less                                2.7.8* Leaner CSS language
meteor-base                         1.0.4* Packages that every Meteor app needs
mobile-experience                   1.0.4  Packages for a great mobile user experience
mongo                               1.1.14* Adaptor for using MongoDB and Minimongo over DDP
msavin:mongol                       2.0.1* In-App MongoDB Editor.. now works better than ever!
pbastowski:angular-babel            1.3.7  Babel compiler and ng-annotate for Meteor 1.3
reactive-var                        1.0.11  Reactive variable
shell-server                        0.2.1* Server-side component of the `meteor shell` command.
standard-minifier-css               1.3.2* Standard css minifier used with Meteor apps by default.
standard-minifier-js                1.2.1* Standard javascript minifiers used with Meteor apps by default.
tracker                             1.1.1* Dependency tracker to allow reactive callbacks
urigo:static-templates              0.1.3  Meteor plugin for importing static HTML templates
zodiase:material-design-icons-fonts  3.0.1  Material Design icons for Meteor
```

Ilustración 6 Árbol de dependencias del proyecto

En esta figura aparece el árbol de dependencias de node. Hay algunas dependencias destacables como meteor, angular, angular material o ui.router, paquete encargado de gestionar la navegación entre pantallas.

```
MacBook-Pro-de-Quique:cassona-webapp quiquesanchebonet$ npm ls
afkar@ /Users/quiquesanchebonet/Desktop/Okyo/cassona-webapp
├── angular@^1.6.2
├── angular-animate@1.5.8
├── angular-aria@1.5.8
├── angular-material@1.1.1
├─┬ angular-meteor@1.3.11
│ ├─┬ jsondiffpatch@0.1.43
│ │ └─┬ chalk@0.5.1
│ │   ├── ansi-styles@1.1.0
│ │   ├── escape-string-regexp@1.0.5
│ │   ├─┬ has-ansi@0.1.0
│ │   │ └── ansi-regex@0.2.1
│ │   ├── strip-ansi@0.3.0
│ │   └── supports-color@0.2.0
│ └── underscore@1.8.3
├─┬ angular-ui-router@0.3.1
│ └── angular@1.5.8 extraneous
├─┬ babel-runtime@6.18.0
│ ├── core-js@2.4.1 extraneous
│ └── regenerator-runtime@0.9.6 extraneous
├── meteor-node-stubs@~0.2.0
├── meteor-rxjs@0.4.7 extraneous
└── rxjs@5.0.3 extraneous
```

Ilustración 7 Árbol de dependencias de Node

4.2.6.Interfaz de usuario web

A continuación, se detallará un esquema que muestra todos los posibles flujos de acción posibles por parte del usuario.
En todas las pantallas se vuelve a la home pinchando en el logo de la barra superior.

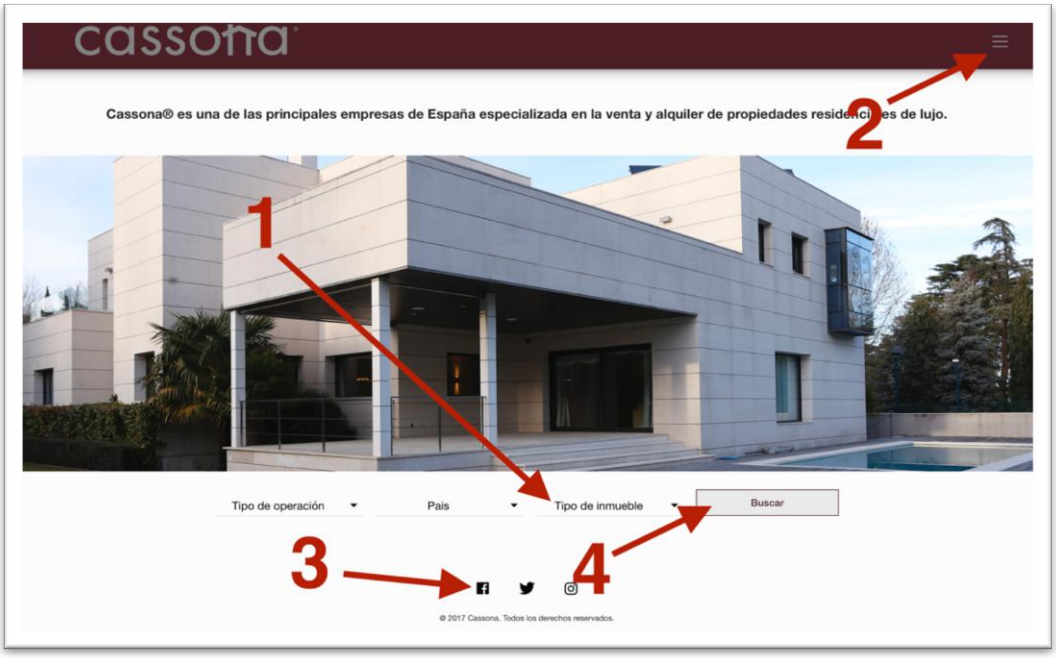


Ilustración 8 Home de la aplicación

Una vez se van escogiendo filtros, estos van ampliándose cada vez mostrando opciones más específicas.

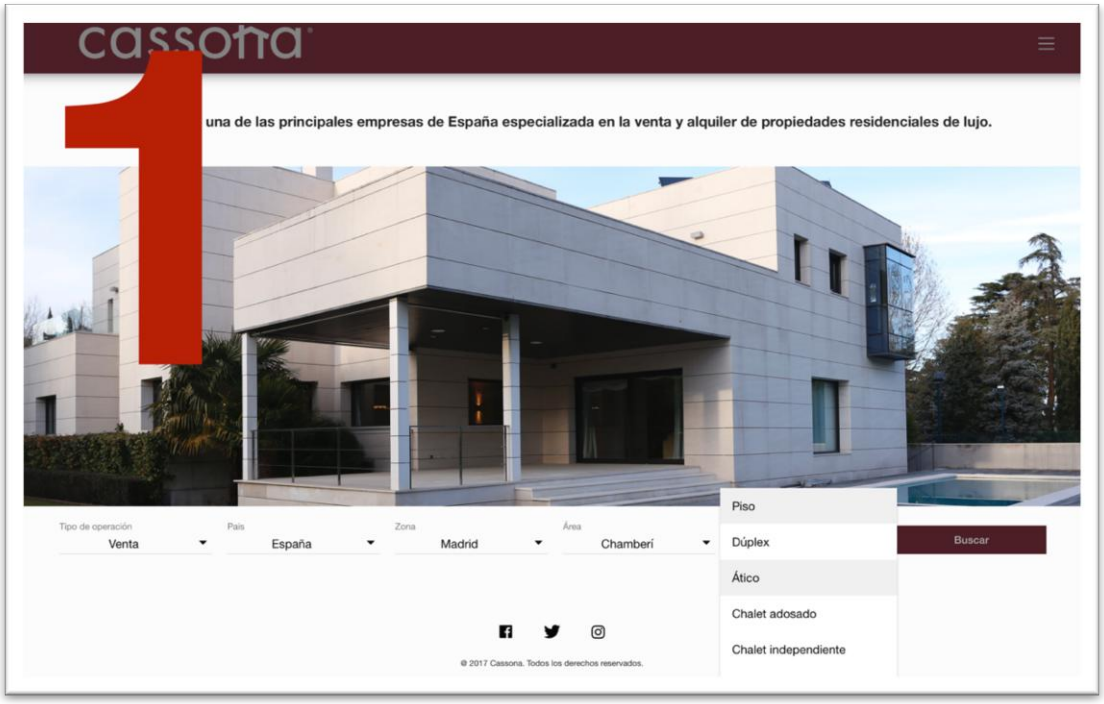


Ilustración 9 Manejo de filtros

La barra desplegable lateral estará disponible desde todas las pantallas.

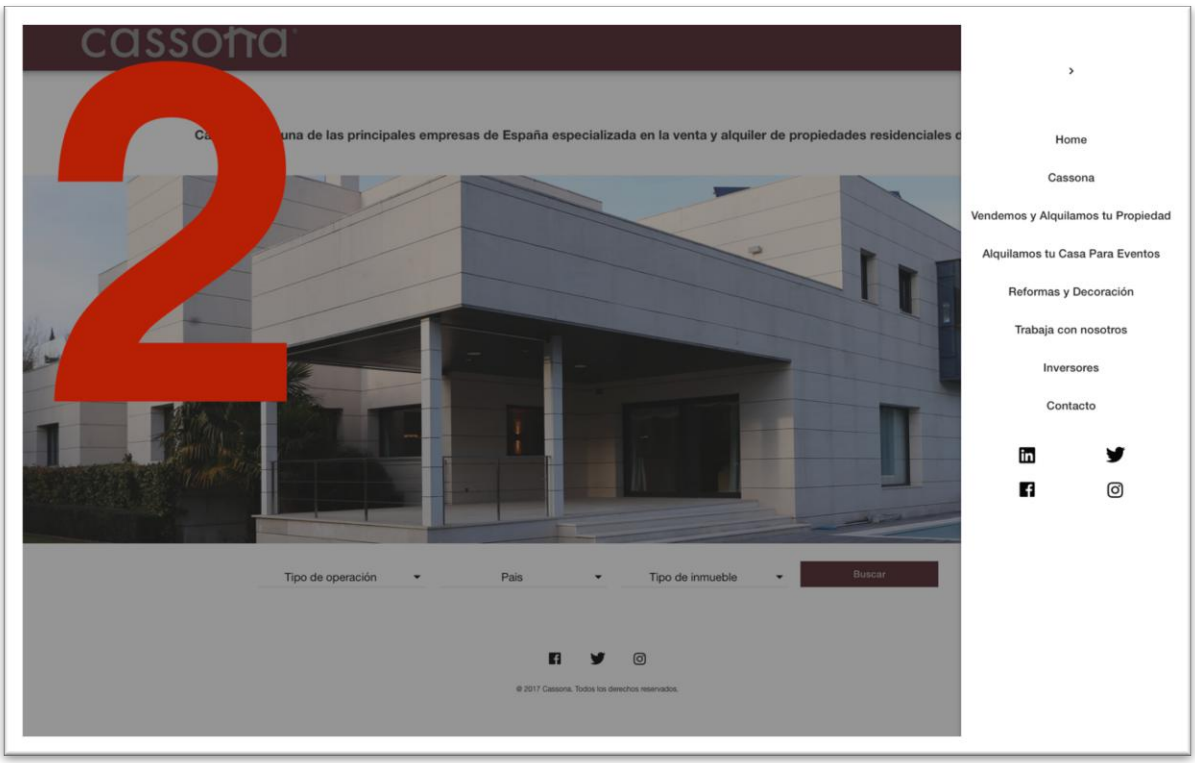


Ilustración 10 Side Bar

Al pulsar en el ítem marcado con un 3 en la primera figura se redirigirá a la red social seleccionada.
Una vez pulsamos en buscar se mostrará un mosaico con las casas que coinciden con la búsqueda.
En caso de no seleccionar ninguna opción se mostrarán todas las propiedades.
El mosaico es completamente responsive y el número de elementos por fila se adaptará de tal forma que se visualicen correctamente en cada pantalla.

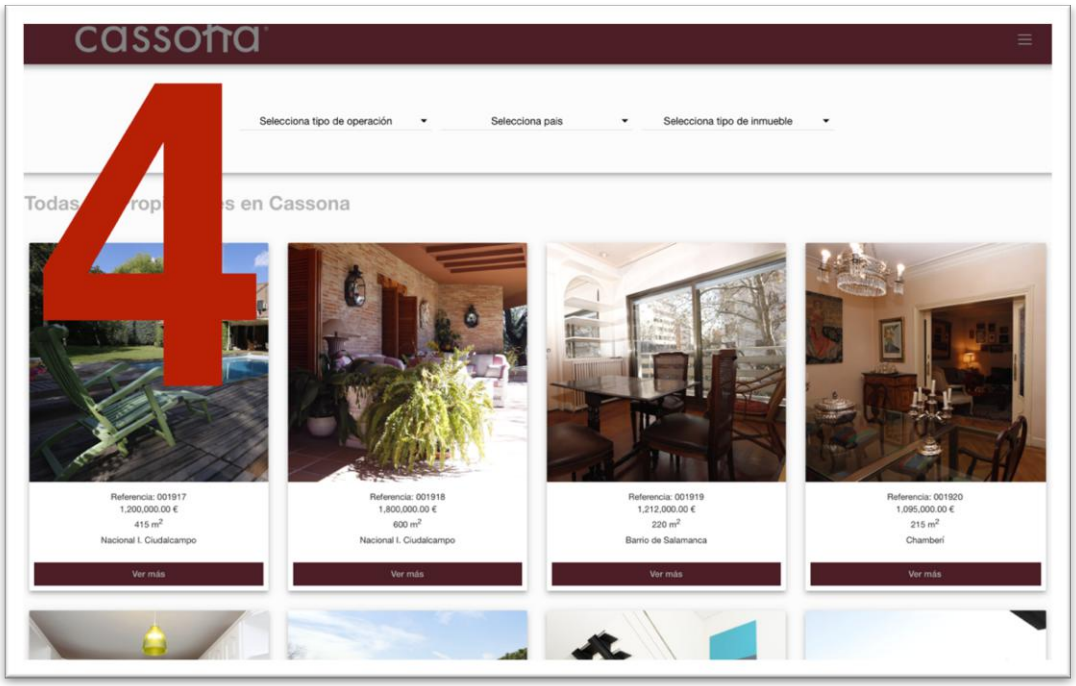


Ilustración 11 Resultados de búsqueda

A continuación, se muestra como aumenta el número de elementos al utilizar una pantalla de mayor tamaño.

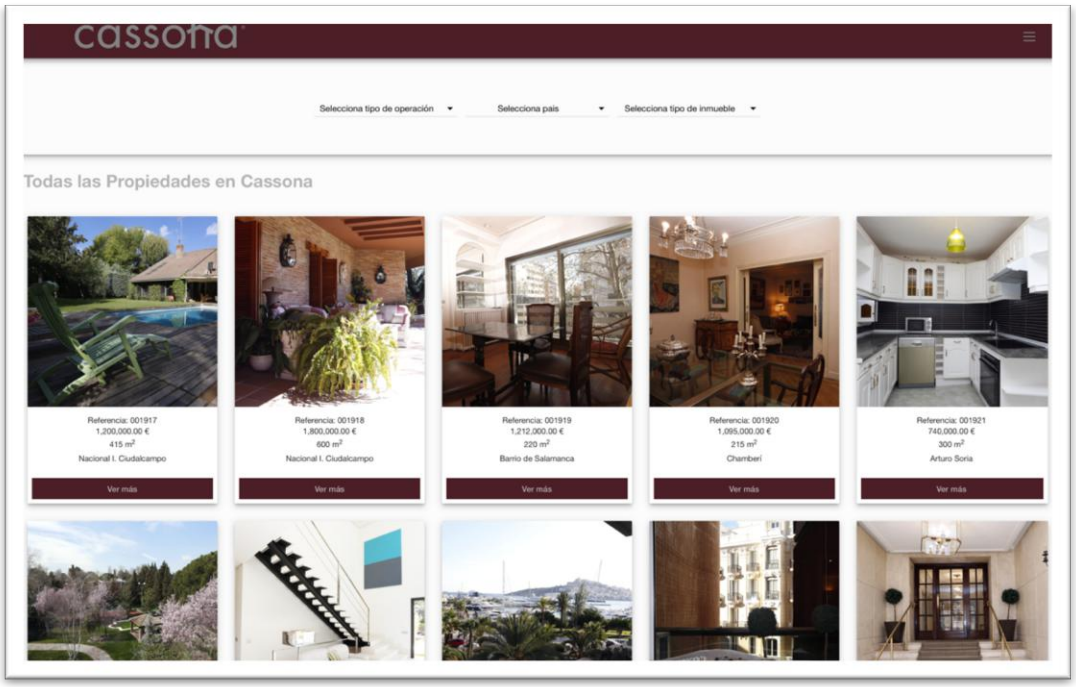


Ilustración 12 Resultados de búsqueda

Una vez se selecciona una miniatura se abre la ficha técnica de la propiedad. Tiene un cuadro de características generales. Una descripción, un botón para solicitar información, un carrusel de fotos y un video. Se muestran algunos ejemplos.

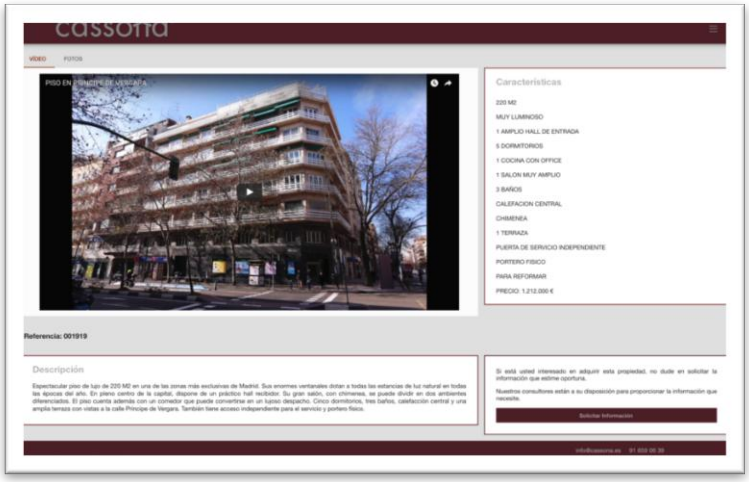


Ilustración 13 Ficha detallada propiedad

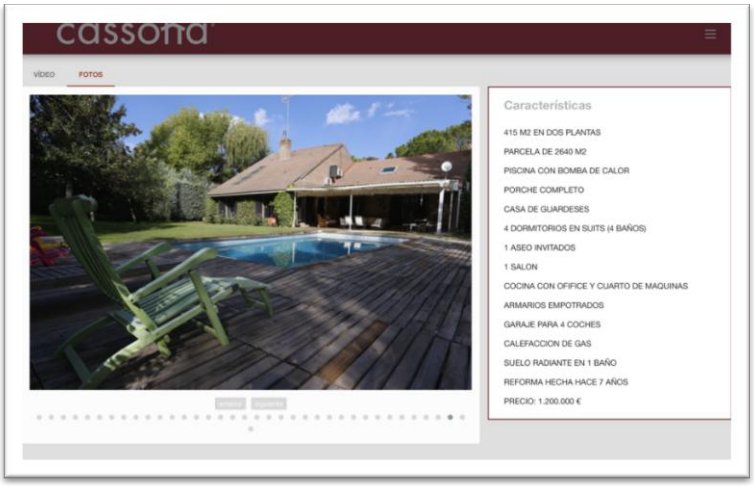


Ilustración 14 Ficha detallada propiedad

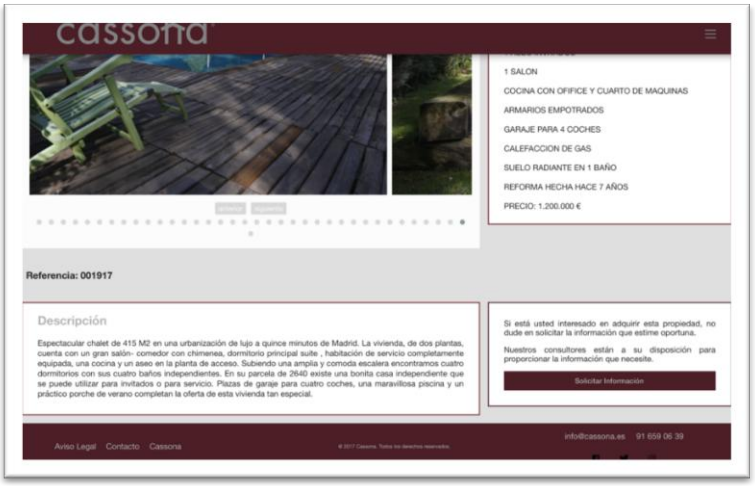


Ilustración 15 Ficha detallada propiedad

Si se pulsa el botón de obtener más información se abrirá el gestor de correo del sistema cliente.

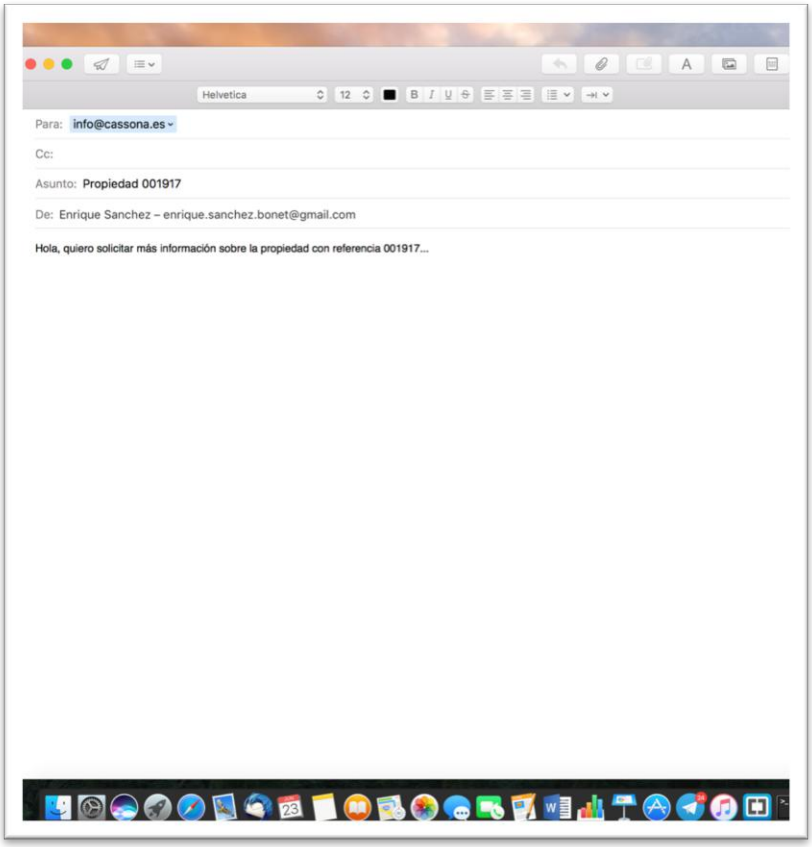


Ilustración 16 Redireccionamiento app correo

Si pulsamos en el aviso legal de la bottom bar en cualquier pantalla se nos redirigirá a la pantalla que muestra dicho mensaje.

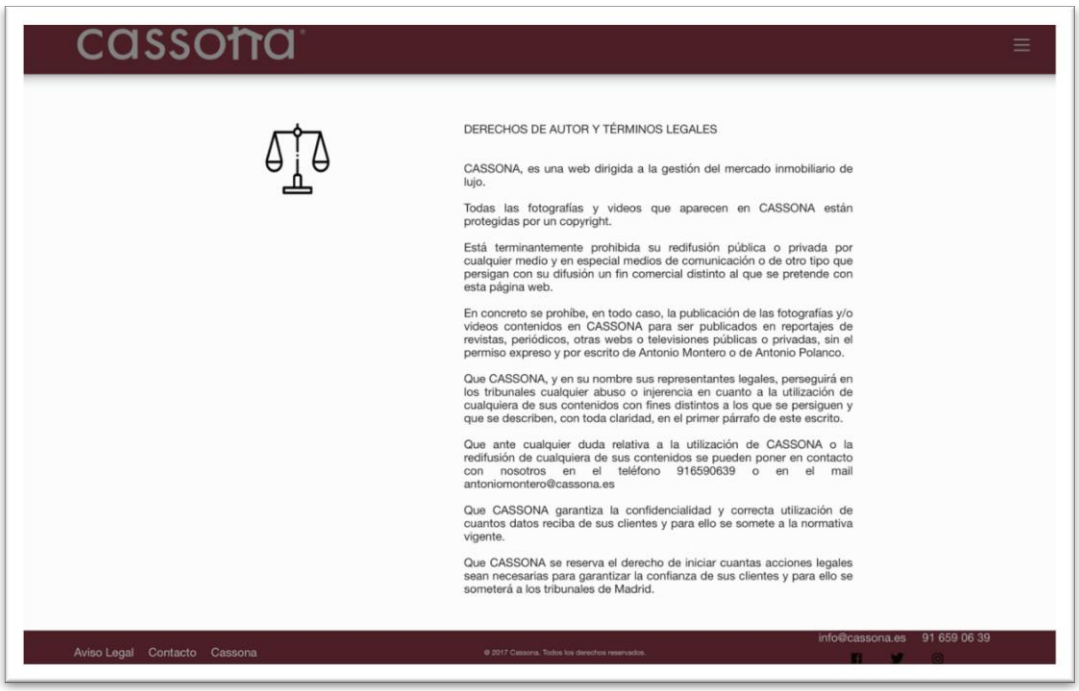


Ilustración 17 Términos legales

Por último, se mostrará un mosaico con las posibles opciones disponibles en la barra lateral desplegable. Completan la oferta de servicios que ofrece la empresa del cliente.

Ilustración 18 Quienes somos

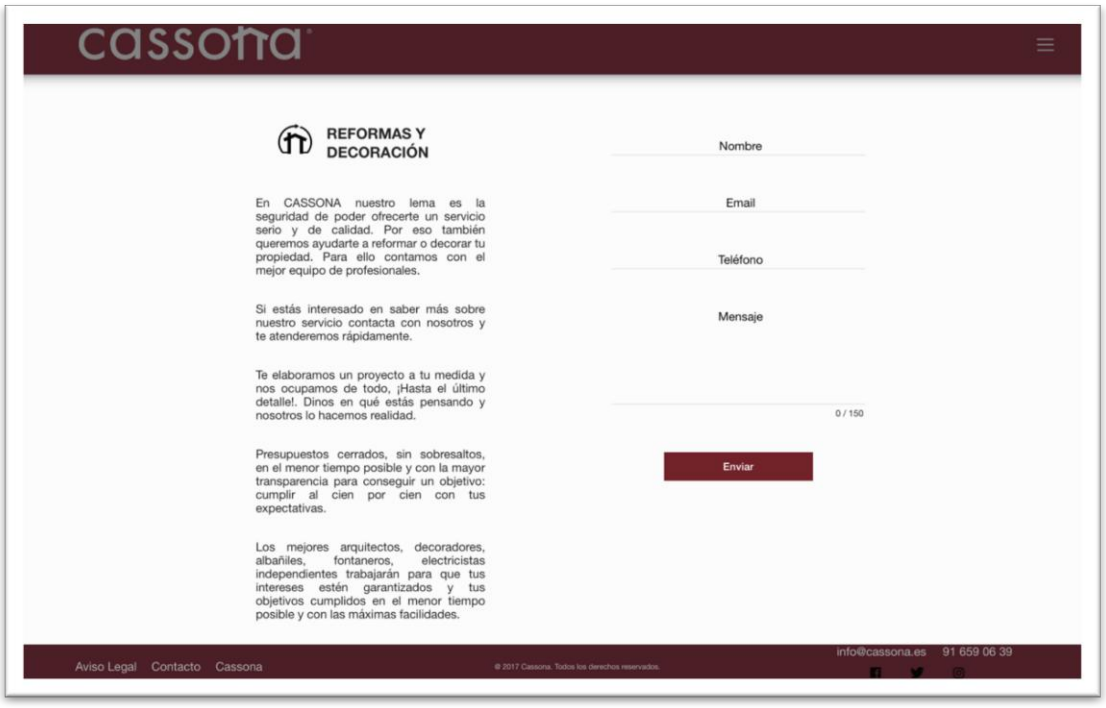


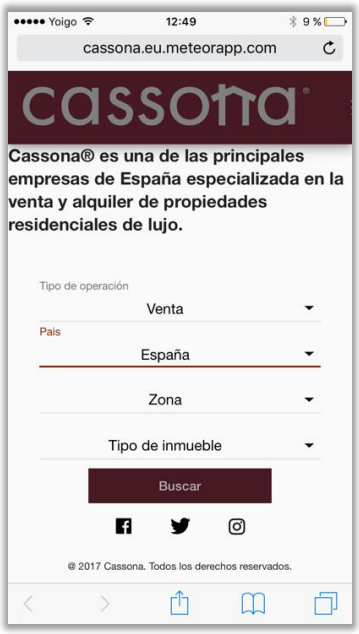
Ilustración 19 Formulario contacto online

El resto de opciones son idénticas a esta última, lo único que cambia es el texto de la izquierda.

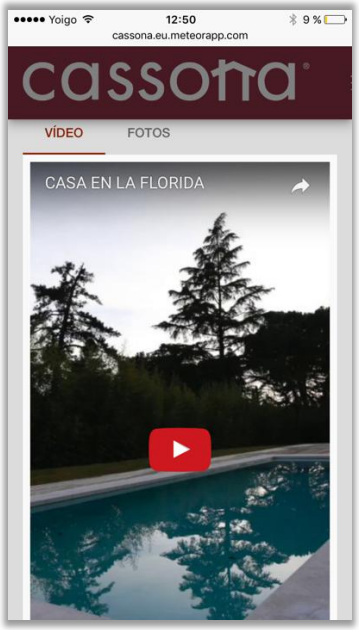
4.2.7.Interfaz de usuario móvil

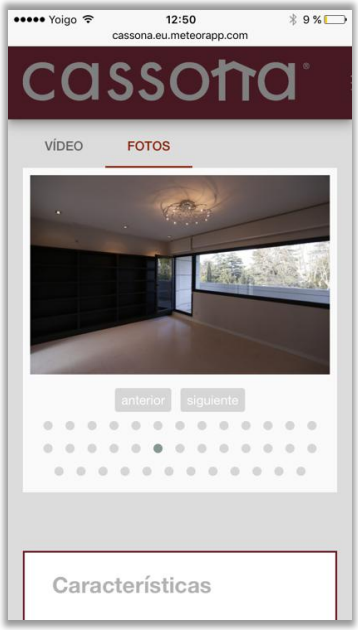
El sistema de navegación es igual en todos los formatos, pero la vista varia significativamente desentendiendo del tamaño de la pantalla del dispositivo. A continuación, se muestra como se visualiza la app en un dispositivo móvil.

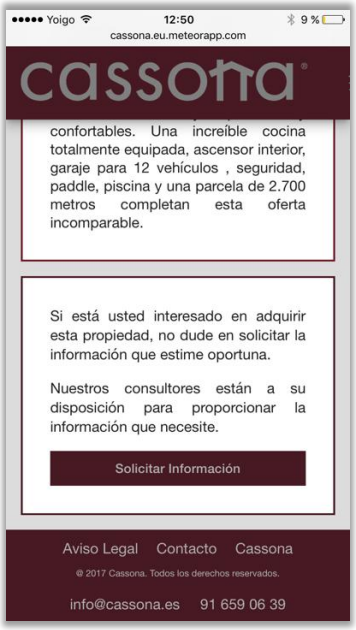
Proceso de búsqueda de una propiedad.



Visualización de una ficha completa





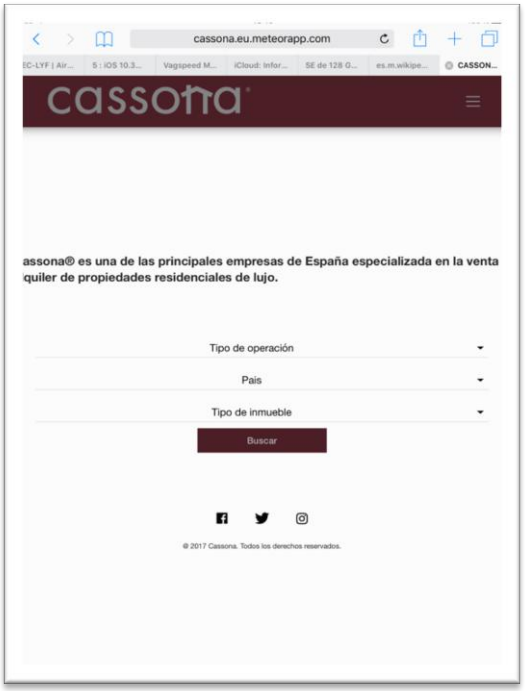


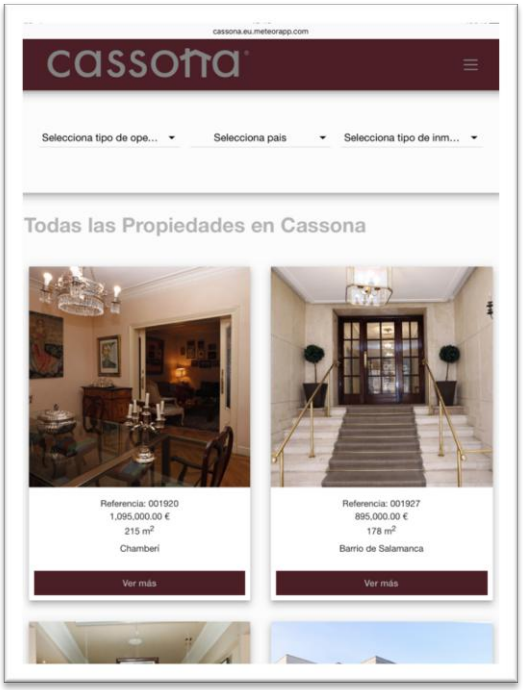
Visualización del resto de funcionalidades, apertura de la app de correo del dispositivo, la sidebar o el formulario de contacto online.

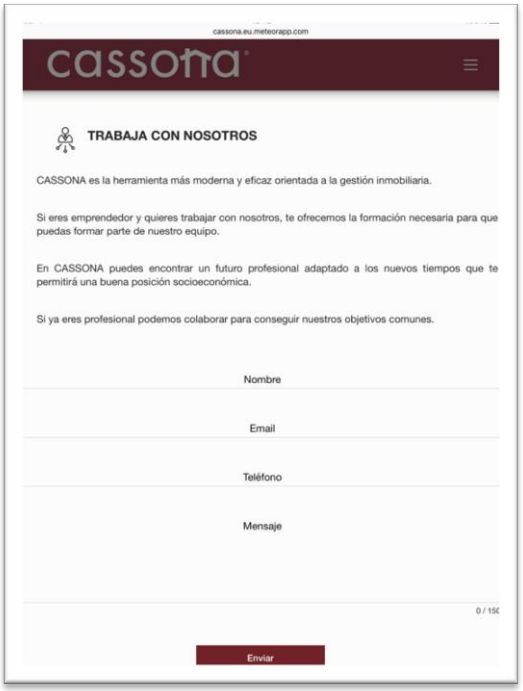
También está optimizada para utilizar el dispositivo en modo apaisado.

4.2.8.Interfaz de usuario tablets y pantallas de tamaño medio.

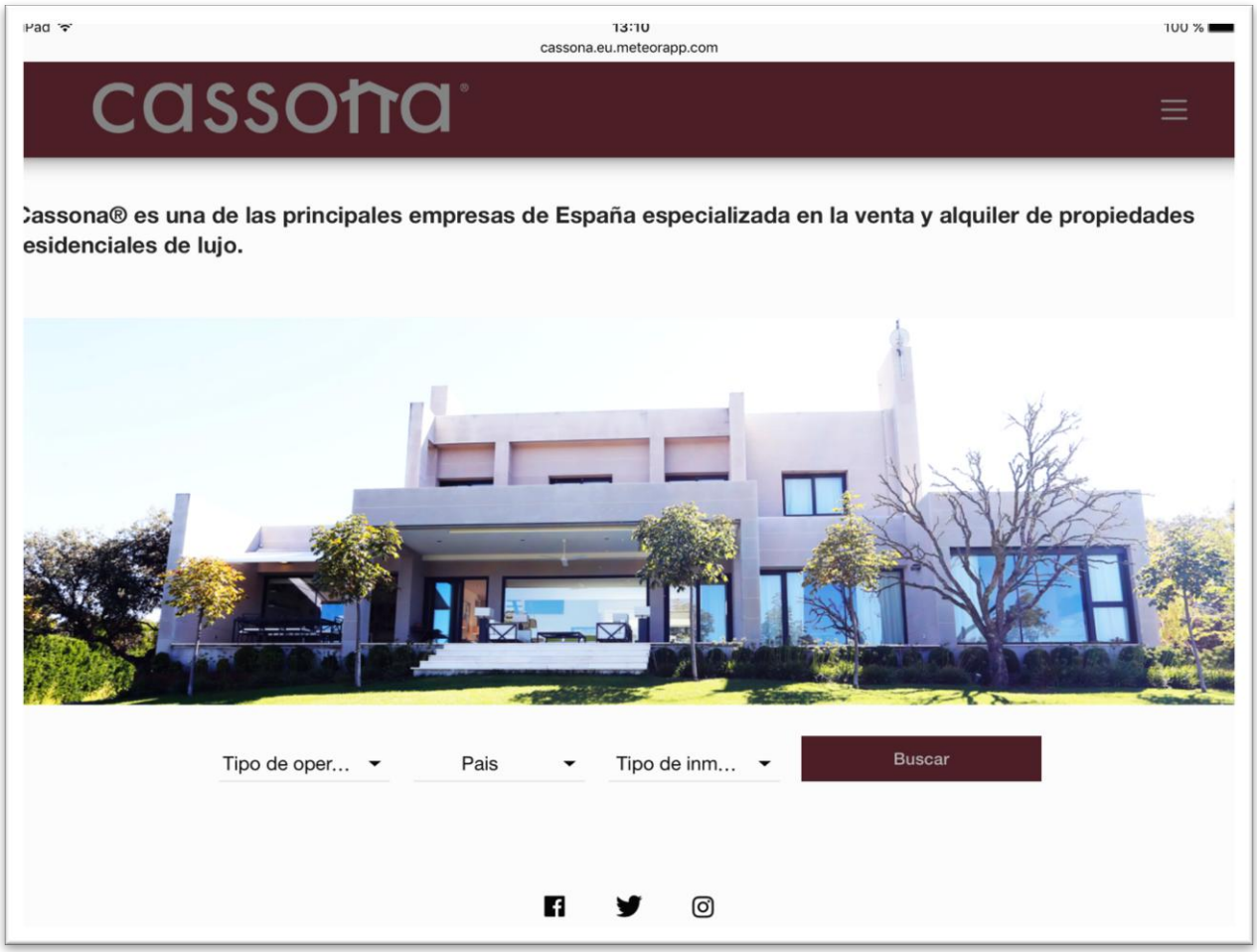
La aplicación también está optimizada para pantallas de tamaño medio como fablets y tablets. A continuación, se muestran una serie de ejemplos.

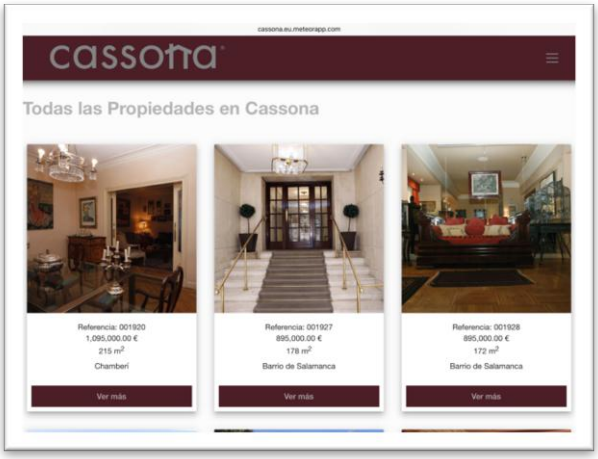


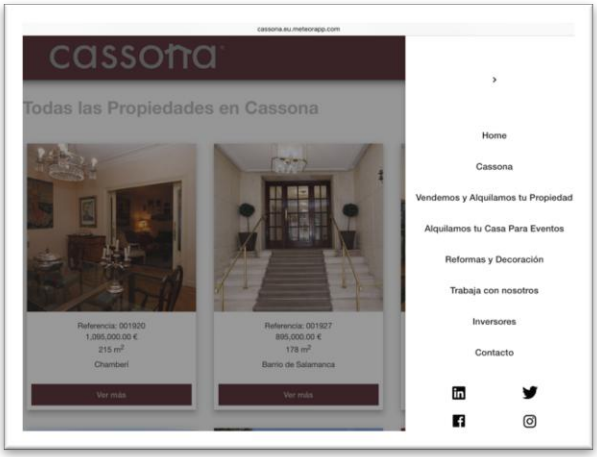


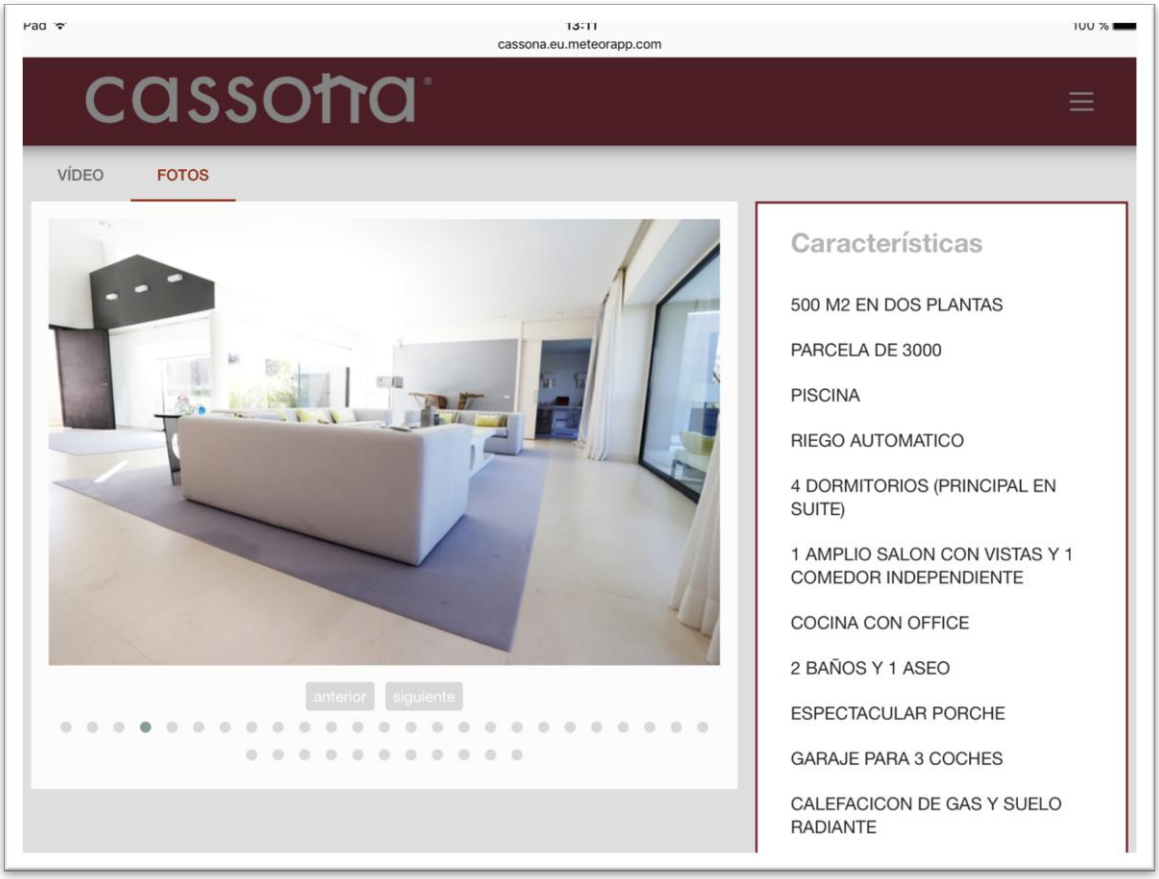


Y al igual que en el caso de los móviles, se ha optimizado la app para modo apaisado. En el caso de las pantallas de tamaño es donde más se puede explotar este modo. En app móvil en cambio es más recomendable usarlo en vertical.








cassona.eu.meteorapp.com
cassona
VÍDEO
FOTOS
anterior
siguiente
Características
500 M2 EN DOS PLANTAS
PARCELA DE 3000
PISCINA
RIEGO AUTOMATICO
4 DORMITORIOS (PRINCIPAL EN SUITE)
1 AMPLIO SALON CON VISTAS Y 1 COMEDOR INDEPENDIENTE
COCINA CON OFFICE
2 BAÑOS Y 1 ASEO
ESPECTACULAR PORCHE
GARAJE PARA 3 COCHES
CALEFACICON DE GAS Y SUELO RADIANTE


cassona.eu.meteorapp.com
cassona
VIGILANCIA 24 H
CONSTRUIDO EN EL AÑO 2009
PRECIO: 1.900.000 €
Referencia: 001923
Descripción
Chalet de lujo de 500 M2 en urbanización exclusiva y en un entorno natural único. La tranquilidad y la independencia le proporcionan un encanto especial a esta magnífica casa con increíbles vistas gracias a sus enormes ventanales por los que la luz del día inunda todas las estancias. Gran salón con comedor independiente, moderna cocina completamente equipada, amplio dormitorio principal tipo suite con fabuloso baño y elegantes vestidores. Despacho y dos dormitorios en la planta superior. Lujoso porche de verano con toldos eléctricos.Garaje para tres coches, piscina y parcela totalmente ajardinada. Para gente sofisticada, visitarla causa una impresión excelente.
Si está usted interesado en adquirir esta propiedad, no dude en solicitar la información que estime oportuna.
Nuestros consultores están a su disposición para proporcionar la información que necesite.
Solicitar Información
Aviso Legal
Contacto
Cassona
@ 2017 Cassona. Todos los derechos reservados.
info@cassona.es
91 659 06 39

5.Integración, pruebas y resultados

Una muy buena práctica, que pocas veces se lleva a cabo, es la implementación de pruebas unitarias durante la fase de codificación. En este proyecto si se han llevado a cabo, ya que, aunque dichas pruebas alarguen el tiempo de la fase de desarrollo, luego la fase pruebas e integración es mucho más rápida y no ocurren problemas inesperados.

JavaScript además es un lenguaje muy permisivo para el programador, por ejemplo, permite no especificar el tipo de los argumentos pasados a una función. Esto puede ser un arma de doble filo, ya que la aplicación no dejará de funcionar en caso de error, pero puede funcionar de forma incorrecta resultando muy complicado identificar el fallo. Por ello el desarrollo de pruebas unitarias es crucial.

Se han desarrollado pruebas utilizando Jasmine, un framework que no tiene dependencia alguna de otras librerías de JavaScript. Se han realizado pruebas en modo standalone, simplemente con un código JavaScript y un HTML [25].

Una vez concluida la fase de desarrollo se han realizado las pruebas necesarias para verificar cada uno de los requisitos funcionales y no funcionales.

5.1.Alcance

En este apartado de especificará el alcance de las pruebas realizadas en los distintos ámbitos requeridos.

5.1.1.Funcionalidad

Las pruebas funcionales han sido un éxito. Se ha logrado cumplir con todos los requisitos especificados por el cliente.
El desarrollo se ha hecho en varias fases incrementales. Según se iban cerrando componentes de la aplicación, tenía lugar una reunión con el cliente donde se analizaba el estado de la aplicación y las mejoras o correcciones que debían hacerse.

5.1.2.Usabilidad

Este apartado también ha sido algo delicado. El cliente en varias ocasiones cambió numerosos aspectos del diseño, buscando mejorar la experiencia de usuario. Finalmente se simplificó la aplicación lo máximo posible, sacrificando opciones que daban más potencia a la aplicación, a cambio de hacer sencilla rápida e intuitiva.
El cliente utilizó una serie de parámetros para medir la calidad de la experiencia del usuario en la aplicación. Se ha medido en todo momento la acción principal de la aplicación, la búsqueda una propiedad con unas características completas.

- Precisión: Mide la cantidad de errores cometidos por el usuario en el proceso de realizar una tare
- Velocidad: Mide el tiempo que tarda el usuario en realizar una tarea concreta. En el caso de esta aplicación, al tener un número limitado de propiedades de muestra, al sujeto de prueba se le encargaba buscar una propiedad con ciertas propiedades predefinidas, ya que

inventándose unos parámetros de cero es probable que no aparezcan resultados.

- Recuerdo: Facilidad que tiene el usuario para repetir un proceso que ha hecho previamente. Este parámetro es el que obtuvo la mejor puntuación de los cuatro. El que según se introducen parámetros de búsqueda haga que se muestren nuevos suele sorprender al usuario y es algo que ya recuerda en las siguientes ocasiones.
- Respuesta emocional: En general la respuesta emocional de la aplicación ha sido muy positiva.

5.1.3.Accesibilidad

La accesibilidad es el único aspecto que no ha dado problemas. AL ser una aplicación web, todas las opciones para mejorar la accesibilidad ya vienen implementadas por los navegadores de escritorio y móvil. Este aspecto ha sido correcto desde el principio.

5.1.4.Compatibilidad

Los requisitos no funcionales son los que más problemas han dado. Al usarse una tecnología tan moderna, es fácil que en navegadores de sistemas antiguos haya cosas que funcionen mal. El cliente probaba la aplicación con todo tipo de sistemas, Windows 7, versiones de Mac antiguas, móviles con un SO muy desfasado etc. También se probaba con versiones totalmente actualizadas.
En un inicio solo se contemplaba que debía funcionar en sistemas modernos y actualizados, por lo que este aspecto ha sido algo delicado en ocasiones. Finalmente se consigo solventar el problema y que todas las partes quedasen satisfechas.

5.2.Desarrollo de las pruebas.

Se ha seguido un modelo tradicional de pruebas. Pruebas de caja blanca en la que se pone a prueba el código, y pruebas de caja negra, donde se asume que solo se conoce la entrada y salida de cada módulo.

Las pruebas de caja blanca son especialmente importantes en este lenguaje de programación. Hay que probar muy bien cada función, cada bucle, cada condición ya que es muy probable que cuando se den situaciones anómalas, el programa continúe funcionando, ya que JavaScript es muy tolerante, pero se den comportamientos extraños los cuales son difíciles de detectar. Por eso es crucial el desarrollo de pruebas de caja blanca durante la fase de codificación.

Las pruebas de caja negra se han realizado en las últimas fases del proyecto, donde simplemente se llevaban al extremo los parámetros de entrada o las respuestas de los módulos independientes, analizando cómo interactúan con el resto de la aplicación.

Las pruebas en las que más énfasis se ha puesto han sido las siguientes:

- Introducir un numero enorme de propiedades (iguales todas) para asegurar que el sistema no se quedaba sin recursos, que el navegador no se sobrecargaba de carga de trabajo y la aplicación seguía siendo fluida.
- El overflow de todos los campos en los que hay texto, de modo que, si en alguna propiedad aparecen datos anómalos, en la vista no se muestra dicho fallo.
- Rellenar formularios de forma incorrecta.
- Imágenes dañadas.
- Numero anómalo de clics.

5.3.Servidor, conexiones y rendimiento

Meteor ofrece la posibilidad de alojar las aplicaciones en sus servidores. Hace que el proceso de subir una aplicación a producción, el cual es tremendamente complejo si se hace del modo tradicional, es bastante sencillo sin dejar de lado la calidad y la variedad de opciones en su configuración, Además, al usar servidores de node, ofrece una capacidad de carga mucho mayor que los hostings habituales a mucho menos dinero.
A continuación, se muestran una serie de imágenes donde se ve claramente la capacidad del servidor para dar servicio a los clientes que se espera a un corto plazo.
En caso de necesitar más potencia bastaría con aumentar el número de contenedores y modificar ciertos parámetros de la configuración como la redundancia etc.
A continuación, se muestran unas imágenes del panel de control de la red de servidores.

Ilustración 20 Monitorización

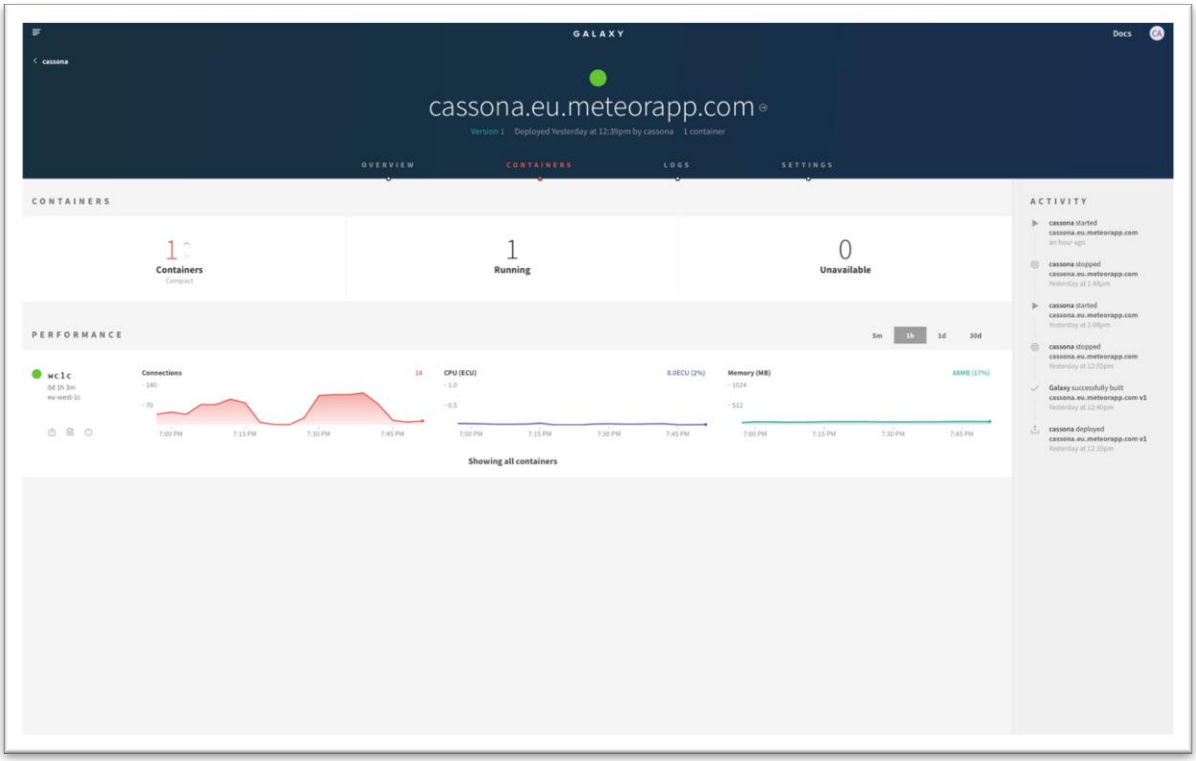


Ilustración 21 Monitorización

5.4.Resultados y conclusiones

Finalmente se ha llegado a una versión aceptable para el cliente. Se han cumplido todos los requisitos funcionales y no funcionales.
Se ha alcanzado la user experience deseada, cumpliendo con creces las métricas impuestas por el cliente.
Se ha entregado una versión robusta frente a fallos, testeada todo lo posible para ahorrar la mayor cantidad de costes en el mantenimiento.
El diseño y apariencia de la aplicación ha obtenido muy buenos resultados por parte del cliente.
Además, se dispone de un sistema de servidores de la mejor calidad, capaz de dar soporte a las expectativas de tráfico a medio plazo y fácilmente ampliable a medida que se vaya registrando un mayor número de usuarios.

6.Conclusiones y trabajo futuro

6.1.Conclusiones

En este trabajo de fin de grado se ha presentado y explicado el proceso de desarrollo de una aplicación a medida para un cliente real.
El campo de la ingeniería informática es a día de hoy uno de los sectores más adecuados para emprender un proyecto empresarial. La web es ya de lejos el medio de comercio más importante y aun muchos negocios no han terminado de adaptarse, hay un nicho de mercado enorme.

Tenemos la suerte de necesitar muy poco capital para echar a andar un proyecto desde cero. Sin embargo, el coste intelectual y de tiempo que se necesita, sobre todo cuando se quiere emprender con poca experiencia, es muy alto.
Gracias a las tecnologías aquí descritas, se pueden emprender proyectos reales, consiguiendo los mismos resultados que una gran consultora que utilice los sistemas tradicionales, pero con una cuarta parte de recursos humanos y de tiempo.

Angular permite desarrollar de forma rápida y sencilla, aplicaciones web de gran complejidad con una calidad excelente. Su gran compatibilidad con todo el ecosistema de node proporciona una potencia y una ventaja competitiva enorme con respecto a otras tecnologías, o bien más arcaicas como HTML puro y php, o más sencillas pero limitadas como un wordpress.

Pero quizás lo que realmente marque la diferencia sea meteor. Nos permite abstraer toda lo relacionado con la rama de sistemas informáticos, la cual es bastante compleja y se necesita de una gran experiencia para poner en producción una aplicación real.
Otro gran problema en este ámbito es alto coste que tienen los servidores tradicionales, que no sueles estar muy optimizados a un gran número de conexiones.
Gracias a meteor podemos olvidarnos de todo lo anterior. Meteor Galaxy nos da solucionada toda la parte de hosting, a un precio muy competitivo y con una gran escalabilidad. Configurar un servidor es tremendamente sencillo y nos permite monitorizar el tráfico y obtener toda la información necesaria para configurar de manera óptima el backend de nuestra aplicación.

El poder abstraer esta parte del proyecto, manteniendo el control de todo lo que sucede, y teniendo la capacidad de configurar y escalar cualquier app, permite a un equipo con poca experiencia centrarse en otras tareas muy necesarias para llevar a buen puerto un proyecto. Solo necesitaremos centrar nuestros esfuerzos en diseñar y desarrollar la aplicación.
Esta ventaja competitiva sin duda permitirá a cualquier equipo de jóvenes emprendedores que quieran iniciarse en el mundo del desarrollo web hacer realidad sus proyectos.

En definitiva, la combinación de angular y meteor permite al desarrollador centrarse en lo importante, teniendo a mano una gran cantidad de herramientas que permiten agilizar el resto de procesos que hay detrás de la propia codificación de la aplicación web.

Por último, destacar la gran importancia que tiene la parte personal, si se quiere llevar a cabo un proyecto similar a este. Las reuniones con el cliente, saber comunicarte con él, entenderle, encontrar el punto medio entre hacer todo lo que te pide y tenerle descontento etc. Son aspectos terriblemente importantes en los que no reparas hasta que te ves metido de lleno en el proyecto. Son cosas de vital importancia para las que, lamentablemente, so se sale preparado de la universidad pero que son las que determinan si el proyecto fracasa o triunfa. Tener los conocimientos técnicos, conocer la tecnología actual, tener el equipo adecuado etc., son cosas que no sirven de nada si no se llevan correctamente las relaciones personales y empresariales con el cliente.

6.2.Trabajo futuro

La idea ultima para este proyecto, es crear un portal inmobiliario de lujo. Para ello ya tenemos una base fuerte, unos buenos cimientos. Tenemos una aplicación robusta, rápida, agradable para el usuario y basada en las últimas tecnologías del desarrollo web.

El siguiente paso es conseguir el posicionamiento. Un buen posicionamiento es clave para destacar. De nada sirve tener la mejor aplicación web jamás hecha si nadie entra.
Por ello, se ha comenzado ya a dar los primeros pasos en esta dirección. Junto con otra compañera especialista en SEO SEM y Marketing digital, se le ha hecho una propuesta al cliente en la que, a lo largo de dos años, se conseguirá poner la aplicación en el top 5 de Google en búsquedas relativas al sector.

Un buen uso de las redes sociales, hacer que la aplicación esté conectada a múltiples plataformas, más la tecnología en la que está desarrollada la aplicación, harán que el posicionamiento SEO suba enormemente, sobre todo después de haber analizado la oferta actual a la que nos enfrentamos. Esto combinado con una pequeña inversión en SEM, pondrán a la aplicación en el lugar que el cliente quiere.

Por otro lado, mientras se consigue lo anterior, se irán añadiendo funcionalidades acordadas con el cliente para el mantenimiento. Empezando por la capacidad de crear usuarios administradores que permiten subir las propiedades sin depender de los desarrolladores, hasta llegar a una plataforma en la que se permita a todo el mundo anunciar sus propiedades.

Es un proyecto ambicioso, pero para el que se cuenta con los recursos personales, técnicos, empresariales y económicos necesarios para hacerlo realidad.

Referencias

[1] «Código Civil art. 334 - Normativa Inmobiliaria». [En línea]. Disponible en: http://normativainmobiliaria.wikidot.com/codigo-civil-art-334. [Accedido: 25-jun-2017].

[2] «Financial Facts: Luxury & Cosmetics in 2016». [En línea]. Disponible en: http://luxurysociety.com/en/articles/2016/12/ey-luxury-and-cosmetics-financial-factbook-2016/. [Accedido: 25-jun-2017].

[3] «Internet es el medio preferido para encontrar casa — idealista/news». [En línea]. Disponible en: https://www.idealista.com/news/inmobiliario/vivienda/2015/12/17/740347-internet-es-el-medio-preferido-para-buscar-casa. [Accedido: 25-jun-2017].

[4] «idealista — Casas y pisos, alquiler y venta. Anuncios gratis». [En línea]. Disponible en: https://www.idealista.com/. [Accedido: 25-jun-2017].

[5] «Fotocasa.es: Alquiler de pisos, compra y venta». [En línea]. Disponible en: http://www.fotocasa.es/es. [Accedido: 25-jun-2017].

[6] «Alquiler de pisos, Venta de pisos en Madrid, Barcelona, Valencia... - Tucasa.com». [En línea]. Disponible en: http://www.tucasa.com/. [Accedido: 25-jun-2017].

[7] «yaencontre: pisos Madrid, pisos Barcelona, pisos alquiler, casas de compra, venta, alquiler y obra nueva». [En línea]. Disponible en: https://www.yaencontre.com/. [Accedido: 25-jun-2017].

[8] «Casas de Lujo, Viviendas Exclusivas, Apartamentos y Chalets de Alto Standing | LuxuryEstate.com». [En línea]. Disponible en: http://es.luxuryestate.com/. [Accedido: 25-jun-2017].

[9] «casas en venta, casas y mansions de lujo, chalets en venta». [En línea]. Disponible en: http://www.luxhome.es/. [Accedido: 25-jun-2017].

[10] «Ambassador - Viviendas Únicas». [En línea]. Disponible en: http://www.ambassador.es/. [Accedido: 25-jun-2017].

[11] «The World Wide Web project». [En línea]. Disponible en: https://www.w3.org/History/19921103-hypertext/hypertext/WWW/TheProject.html. [Accedido: 25-jun-2017].

[12] Natalia Arroyo Vázquez, «¿Web 2.0? ¿web social? ¿qué es eso?»

[13] «AngularJS — Superheroic JavaScript MVW Framework». [En línea]. Disponible en: https://angularjs.org/. [Accedido: 24-jun-2017].

[14] «Bootstrap · The world’s most popular mobile-first and responsive front-end framework.» [En línea]. Disponible en: http://getbootstrap.com/. [Accedido: 24-jun-2017].

[15] «React - A JavaScript library for building user interfaces». [En línea]. Disponible en: https://facebook.github.io/react/. [Accedido: 23-jun-2017].

[16] «Ember.js - A framework for creating ambitious web applications.» [En línea]. Disponible en: https://emberjs.com/. [Accedido: 23-jun-2017].

[17] «Sass: Syntactically Awesome Style Sheets». [En línea]. Disponible en: http://sass-lang.com/. [Accedido: 21-jun-2017].

[18] «Getting started | Less.js». [En línea]. Disponible en: http://lesscss.org/. [Accedido: 24-jun-2017].

[19] «Node.js». [En línea]. Disponible en: https://nodejs.org/es/. [Accedido: 22-jun-

2017].
[20] «Build Apps with JavaScript | Meteor». [En línea]. Disponible en: https://www.meteor.com/. [Accedido: 23-jun-2017].
[21] «Reinventando la gestión de datos | MongoDB». [En línea]. Disponible en: https://www.mongodb.com/es. [Accedido: 22-jun-2017].
[22] «Diseño gráfico plano o flat design». [En línea]. Disponible en: http://www.mique.es/diseno-grafico-plano-o-flat-design/. [Accedido: 22-jun-2017].
[23] «How to use Model-View-Controller (MVC)». [En línea]. Disponible en: http://web.archive.org/web/20150518095937/http://st-www.cs.illinois.edu:80/users/smarch/st-docs/mvc.html. [Accedido: 21-jun-2017].
[24] «Stack MEAN. Introducción a Mongo, Express, Angular y Node». [En línea]. Disponible en: http://funnyfrontend.com/introduccion-stack-mean-parte-1/. [Accedido: 22-jun-2017].
[25] «Jasmine Documentation». [En línea]. Disponible en: https://jasmine.github.io/. [Accedido: 20-jun-2017].

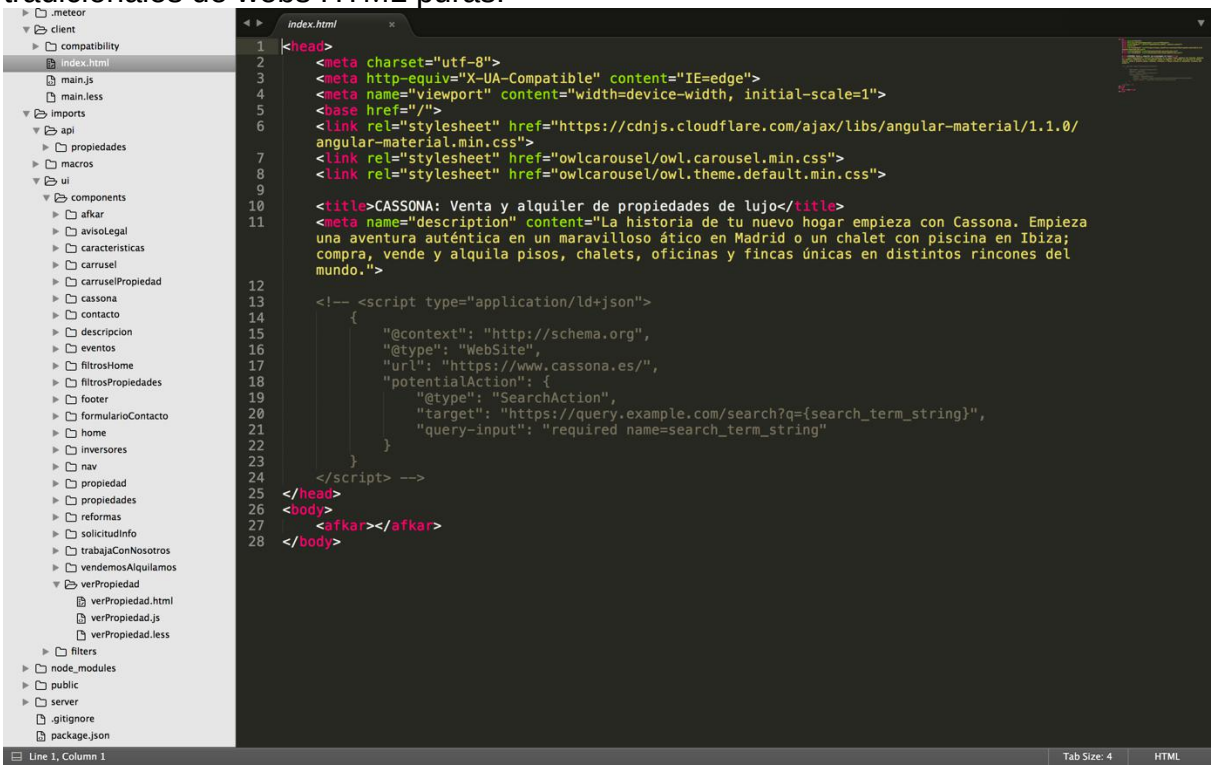
Glosario

API	Application Programming Interface.
JS	JavaScript.
DB	Base de datos.
Framework	Conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.
Deploy	Desplegar una aplicación web para producción.
UEX	Experiencia de usuario.
SEO	Posicionamiento en Google basado en sus algoritmos.
SEM	Posicionamiento en Google de pago.

Anexos

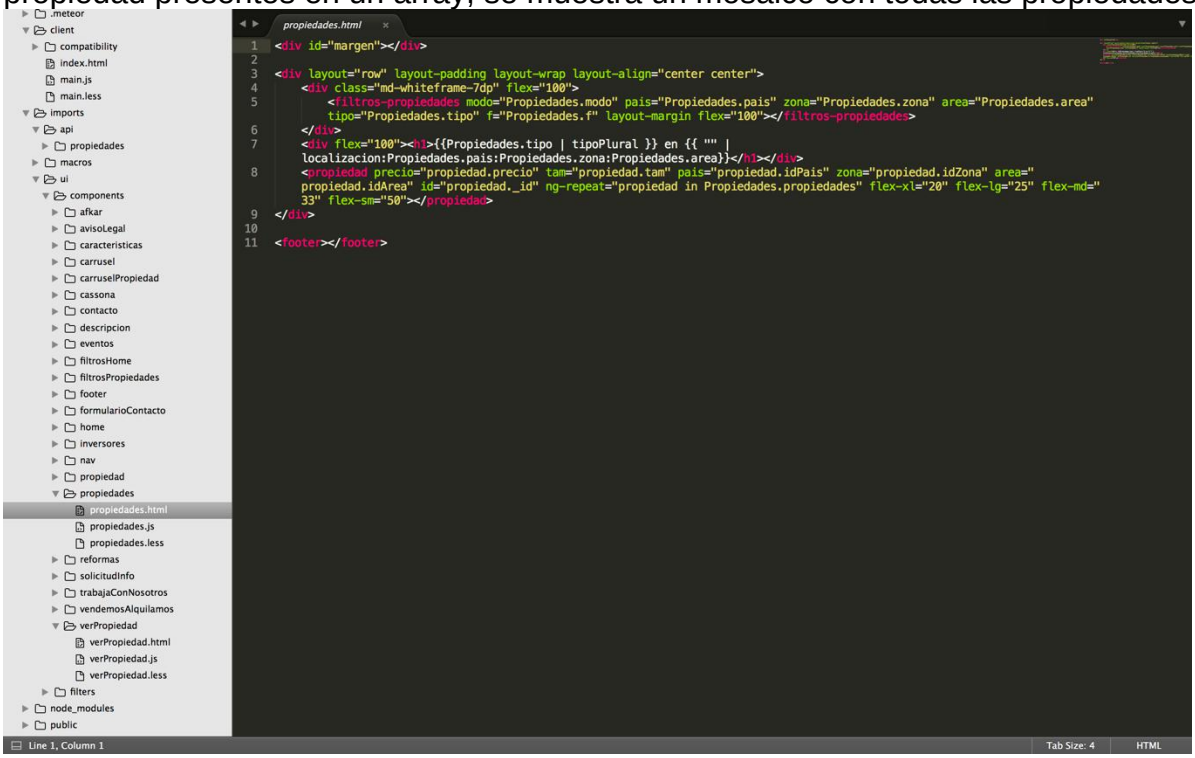
A. Index de la aplicación.

A continuación, se puede ver en la barra lateral izquierda la estructura del proyecto. Se ve claramente la modularidad de este gracias a la multitud de componentes diferentes.
Se puede apreciar también la simplicidad del index HTML, comparado con los tradicionales de webs HTML puras.



B. Ejemplo de componente Angular

En este caso se muestra el componente propiedades, que se encarga de mostrar el mosaico de propiedades resultantes de una búsqueda.
Destacar el ng-repeat que se encarga de mostrar la misma vista HTML por cada elemento en un array del controlador y el tag propiedad que es el componente que muestra la miniatura de una casa.
De este modo, al llamar al componente propiedad pasándole los datos de cada propiedad presentes en un array, se muestra un mosaico con todas las propiedades.



C. Presupuesto SEO/SEM

PRESUPUESTO SEO/SEM

-Google Analytics	
- Analítica web	**250**
-Informe trimestral	**250**
-Estudio de las palabras clave	**1.000**
- Estudio e investigación competencia.	
- Análisis de patrones de búsqueda.	
-Monitorización e informes	**700**
- Reporting de evolución de las posiciones, KPIs, backlinks propios….	
-SMO (Social Media Optimization)	**300**
-Documento de recomendaciones	**500**
-Rastreo de páginas y análisis de indexión	
-Campañas en Google Adwords	
Honorarios	**1.000**
Inversión*	
Testeo	**200**
Estimación campañas**	**2.200****
-Gastos de consultoría	**400**
TOTAL HONORARIOS	4.400
COSTES DIRECTOS SEO/SEM	2.400**
TOTAL PRESUPUESTO	**6.800****

- (*)Estimado, en función de resultados podría variar según el testeo cuya duración será de un mes.
- (**) Los 11 meses restantes, estimamos 200e por mes, podría variar en función de los resultados del testeo. Limitando la inversión mensual según convenga el cliente.